## ANSCHUETZ/WEISGERBER/ANSCHUETZ

# ATARI 8-BIT BASIC GAMES




**Written by Robert & Eric Anschuetz and John Weisgerber**

**Version 4.0**

**April 20, 2018**

## CHANGE LOG

## Version 4.0 (April 20, 2018)

- Added link to YouTube video discussing A/W/A games.

## Version 3.0 (August 25, 2017)

- Changed Title
- Added Change Log
- Changed Title Page Cover Art to Show the A/W/A Team's Hardware
- Renamed A/W/A Development Notes Section to A/W/A Game Write-Ups
- Enhanced Several Descriptions in the A/W/A Basic Games Section
- Added Atarimania.com Hyperlinks to the A/W/A Basic Games Section
- Added Several Letters to the A/W/A Magazine Correspondence Section
- Added Several Games to the A/W/A Game Write-Ups Section
    - Alien Assault
    - Kooky Diver
    - Robot Dungeon
    - Space Assailants (Several Pages Added to the Initial Write-Up)
- Added Several Games to the A/W/A Basic Games Section
    - EMU Track Meet Database
    - Phoenix
    - Piracy
    - Ramses' Revenge
- Added Several Games to the A/W/A Magazine Correspondence Section
    - Kooky Diver
    - Piracy II
    - Robot Dungeon
- Added Source Code Listings for Several Games
    - EMU Track Meet Database
    - Phoenix
    - Piracy
    - Ramses' Revenge

## Version 2.0 (August 15, 2017)

- Changed Cover Art
- Updated Anschuetz/Weisgerber/Anschuetz Section
- Added A/W/A Development Notes Section
    - Kooky Klimber
    - Kooky's Quest

    - Night Rescue
    - Overflow
    - Space Assailants
- Added A/W/A Magazine Correspondence Section
    - Kooky Klimber
    - Kooky's Quest
    - Night Rescue
    - Overflow
    - Space Assailants
- Added Anschuetz High Score / Game Lister to A/W/A Basic Games Section
- Added Anschuetz High Score / Game Lister to A/W/A Basic Code Listings Section

## Version 1.0 (August 2, 2017)

- Initial Release

# TABLE OF CONTENTS


ANSCHUETZ/WEISGERBER/ANSCHUETZ ..... 1

EARLY DAYS ..... 1

PUBLISHED GAMES ..... 3

PHREAKS AND PIRATES ..... 4

GAME OVER ..... 5

LASTING LEGACY ..... 6

A/W/A BASIC GAMES ..... 8

ANSCHUETZ HIGH SCORE / GAME LISTER (1985) ..... 9

ALIEN ASSAULT (1983) ..... 11

EMU TRACK MEET DATABASE (1987) ..... 15

KOOKY DIVER (1986) ..... 18

KOOKY KLIMBER (1984) ..... 22

KOOKY'S QUEST (1985) ..... 26

NIGHT RESCUE (1984) ..... 31

OVERFLOW (1985) ..... 35

PHOENIX (1982) ..... 38

PIRACY (1982) ..... 40

PIRACY II (1986) ..... 41

RAMSES' REVENGE (1983) ..... 44

ROBOT DUNGEON (1985) ..... 46

SOKO-BAN (1988) ..... 54

SPACE ASSAILANTS (1986) ..... 57

YAHTZEE (1987) ..... 60

A/W/A GAME WRITE-UPS ..... 62

ALIEN ASSAULT ..... 63

KOOKY DIVER ..... 72

KOOKY KLIMBER ..... 79

KOOKY'S QUEST ..... 89

NIGHT RESCUE ..... 93

OVERFLOW ..... 98

ROBOT DUNGEON ..... 103
SPACE ASSAILANTS ..... 109
A/W/A MAGAZINE CORRESPONDENCE ..... 121
KOOKY DIVER ..... 122
KOOKY KLIMBER ..... 127
KOOKY'S QUEST ..... 133
NIGHT RESCUE ..... 136
OVERFLOW ..... 141
PIRACY II ..... 145
ROBOT DUNGEON ..... 148
SPACE ASSAILANTS ..... 151
A/W/A PUBLISHED GAMES ..... 157
KOOKY'S QUEST ..... 158
OVERFLOW ..... 166
ROBOT DUNGEON ..... 170
A/W/A BASIC CODE LISTINGS ..... 172
ANSCHUETZ HIGH SCORE / GAME LISTER ..... 173
ALIEN ASSAULT ..... 174
ALIEN ASSAULT - 2017 ..... 175
EMU TRACK MEET DATABASE ..... 176
EMU TRACK MEET DATABASE – AFFILIATIONS ..... 177
EMU TRACK MEET DATABASE – CREATE HEAT ..... 179
EMU TRACK MEET DATABASE – EVENTS ..... 180
EMU TRACK MEET DATABASE – PARTICIPANTS ..... 182
EMU TRACK MEET DATABASE – RESULTS ..... 184
EMU TRACK MEET DATABASE – START NEW MEET ..... 186
KOOKY DIVER ..... 187
KOOKY KLIMBER ..... 189
KOOKY'S QUEST ..... 191
NIGHT RESCUE ..... 194
OVERFLOW ..... 196
PHOENIX ..... 197

PIRACY ..... 198
PIRACY II ..... 200
RAMSES' REVENGE ..... 203
ROBOT DUNGEON ..... 204
ROBOT DUNGEON – 2017 ..... 207
ROBOT DUNGEON – DUNGEON CREATOR ..... 210
ROBOT DUNGEON – DUNGEON CREATOR - 2017 ..... 211
ROBOT DUNGEON – ROBOHELP ..... 212
SOKO-BAN ..... 213
SOKO-BAN – SAVDAT ..... 216
SPACE ASSAILANTS ..... 217
YAHTZEE ..... 218

# ANSCHUETZ/WEISGERBER/ANSCHUETZ

## EARLY DAYS

The beginning of the Anschuetz/Weisgerber/Anschuetz (AWA) team's obsession with the Atari 8-Bit Computer system began on Christmas Day, 1981, when Eric and Robert Anschuetz received an Atari 400 for Christmas along with a 410 Cassette Recorder for data storage and a BASIC Programming Language Cartridge. The Atari 400 computer hooked up to their small 13" Color TV in their bedroom. The Atari 400 came with a user's manual and a BASIC programming guide. Shortly after they got their computer, they upgraded it from the standard 16K memory to 48K. Now they had a real computer! Along with the Atari 400, they soon purchased the games Star Raiders, Missile Command, Ghost Hunter (a Pac-Man clone), and Galactic Chase (a Galaxian clone). Eric and Robert also obtained an Atari Programmer's Manual that unlocked some of the secrets of the computer and provided several programming examples. This manual was thoroughly read and referenced often over the years.

Since Eric and Robert weren't experienced at typing, the very Christmas Day that they received the computer they had their Dad type in a BASIC program called "Shoot" from the September 1981 issue of COMPUTE! Magazine. "Shoot" was similar to Air-Sea Battle on the Atari 2600 VCS, and was actually a machine-language program that used a BASIC listing to call the machine language routines. This BASIC language listing for "Shoot" opened their eyes to the world of programming and the tricks necessary on the Atari for displaying colorful images, making sounds, and making things move with a joystick.

Eric and Robert's best friend, John Weisgerber, shared their passion for video games and their curiosity to find out how to write programs for their computer. John paid almost daily visits to their house to play games and type in programs written in BASIC from magazines. Each program they typed in was another learning experience. There were three primary magazines that catered to the Atari 8Bit (400 and 800) computers: A.N.A.L.O.G., ANTIC, and COMPUTE!. Two other magazines, SoftSide and Creative Computing, mostly covered the Apple II, but occasionally published Atari-related articles and type-in games.

Eric, Robert, and John spent the summers in Ypsilanti, Michigan taking long walks and riding their bikes. They often walked 10 miles or so each way to Briarwood Mall in Ann Arbor, although more often they would bike there. They so looked forward to receiving the latest issue of A.N.A.L.O.G, Antic, and COMPUTE!, that they knew the exact date that the magazines would arrive at the B. Dalton book store in Briarwood Mall and often timed their bike rides and walks to arrive on the same days that the magazines did. On more than one occasion, they had to ask the store manager to check in the back of the store for the day's arrival of magazines and got the latest issue straight out of the box.

It didn't take long until the A/W/A Team started writing their own games. The first game that any of them wrote was a game called "Phoenix" written by Robert. Phoenix was semi-playable and was, of course, modeled after the arcade game Phoenix. Probably the first fully playable game was a two-player game that took place in a pyramid called Ramses' Revenge. The A/W/A Team spent many late

nights drawing game characters and background graphics on graph paper, and then encoding them within numerical values that represented the 1's and 0's that comprised the bitmap of the characters. Ramses' Revenge was especially noteworthy in that Robert, Eric, and John discovered how to program and use a redefined character set the same day that they programmed the game. They studied code from other programs in the Atari magazines and within a day had learned the concept of redefining character sets and developed a complete game using the technique!

Another early game that the A/W/A Team worked on was a racing game based on the Atari arcade game Night Driver. This game was called "Detroit Grand Prix," because the driving course in the game was an actual representation of the Detroit Grand Prix race course, which was held on the streets near the Detroit riverfront around the Renaissance Center. The A/W/A Team took a map of the Detroit Grand Prix track, and encoded it in such a way that it would be represented within the game. The visuals for the game placed a static nose of a race car at the bottom of the screen. On the nose of the car, the #14 and the word "Foyt" was written upside down, as it would be seen from the driver's perspective. This was accomplished using a unique character mode of the Atari that flipped text upside down. The car navigated through pylons that came toward the driver, and the object was to remain in the middle of the pylons without crashing into them.

Still another game the A/W/A Team wrote was a game patterned after Midway's Space Encounters. This game took place in a trench similar to the death star trench scene in Star Wars. The sides of the trench rotated through colors, giving the illusion that your ship was moving forward. The A/W/A Team programmed the ship to move around in accordance with the joystick movement, but development stopped at that point due to reaching some sort of technical limitation of their Atari programming knowledge. They never got the enemy ships programmed and ended development of the game.

These early games were developed using the Atari 410 cassette recorder as a backup device since the A/W/A Team didn't yet own a disk drive. The 410 was very unreliable, and it was only a 50/50 proposition whether the saved data could be re-loaded the next day. For every two steps forward in development, it seemed like there was always a step backwards with lost data. Unfortunately, because of the unreliability of the Atari 410 cassette recorder, some of these games including Detroit Grand Prix and the Space Encounters game, were lost to nothing but memory. Robert and Eric bought a degaussing wand that they plugged in and waved in front of the read/write heads of the cassette recorder. Presumably, this made the read/write functions of the cassette recorder a little bit more reliable, but not by much. Eric and Robert didn't have a printer in those days to even print out the "listing" of the game as a backup in case the cassette recorder failed.

Because of the very limited RAM available, the A/W/A Team was very conscious of memory saving techniques. Their modified Atari 400 had 48K of RAM, but most games needed to be written for 16K so that publishers could be assured that every Atari owner could run the games. They found out that every time a number was referenced, it took six bytes to store the floating-point version of the number, even if it was an integer, like 0! However, Atari BASIC used a clever way of tokenizing source code so that the first time a variable was used, it took something like six bytes, but every time it was referenced after that only took a byte or two. Because of this, many Atari BASIC programs had code at the beginning like

ZERO = 0 and then used references to the variable ZERO throughout the code rather than using the six byte "0" in order to save code. In order to give lasting tribute to themselves and friends, they would almost always use their initials as variables in the BASIC code (e.g. EA = 1, RA = 0, and JW = 100). It was a nice tribute, but it led to very unreadable code! In addition, comment statements (REM in BASIC), would take a byte for each character in the comment. They couldn't afford any extra memory, so the comments were the first to go. Again, this made their code very difficult to read. They would almost always make these changes at the very end of coding so that it was not so difficult to read as they were developing, but the finished product may just as well have been encrypted!

# PUBLISHED GAMES

Those early games like Phoenix, Ramses' Revenge, Detroit Grand Prix, and Space Encounters were a great learning experience for developing even better games. At the same time, more and more information about the secrets of Atari Player/Missile Graphics, Display Lists, Redefined Character Sets, Hidden Graphics Modes, Assembly Routines, Sound Effects, and Horizontal/Vertical Scrolling were being documented in the Atari magazines. Combining their experience with the new-found secrets of programming led the A/W/A Team to pursue writing BASIC games that were worthy of publication in one of the Atari-based magazines.

The A/W/A Team worked late hours in Robert and Eric's bedroom. Most of the productivity occurred in the summer when they were out of school and could spend lots of times designing and playing games. They worked well as a team and never argued over game development ideas. They took turns at the keyboard, and while one person was typing in the code, the others were writing code out on paper or designing graphics on graph paper. The A/W/A Team didn't have a printer at that time, so all source code was typed directly into the ATARI BASIC editor directly from notes scratched out on paper. They learned to back-up their BASIC programs on cassette often, but often not enough to prevent losing lots of work.

Taking all of the information that they had learned, the A/W/A Team started writing games like Kooky Klimber, Night Rescue, and Kooky's Quest. These games are covered in detail below. The A/W/A Team added a standardized 3-line title screen centered on the screen to all of their programs. The title screen mixed text modes and each line had text of a different size and color, contrasted typically against a black background. The standardized A/W/A game title was listed using large text on the first line, the second line was a status line (typically either "Initializing" or "Press START to Begin"), and the third line proudly displayed the name that the team went by "Anschuetz/Weisgerber/Anschuetz." After showing the first game that implemented this title screen to Robert and Eric's Dad, who was an English Major and professional editor and copywriter, their Dad said "'Initializing' is not a word!"

Kooky Klimber was the first game that the A/W/A submitted to a magazine, and it was purchased by COMPUTE! magazine for $400. Shortly after that, Night Rescue was written and sold to COMPUTE! for another $400. Unfortunately, both of these games were never published. Kooky's Quest, Overflow,

and Robot Dungeon were sold to Antic magazine for slightly less money ($200-$300), and they were ultimately published in 1985.

About a year or two after Robert and Eric got their Atari 400, John Weisgerber worked on his parents and got an Atari 400 as well. With some of the earnings from selling games to magazines, Robert and Eric later upgraded to an Atari 800XL with built-in BASIC and a full-keyboard. Along with the Atari 800XL, they also bought an Atari 1050 Disk Drive, a color Okimate 10 printer, an MPP 1000C 300-baud modem, and a small 4-color miniature plotter known as the Atari 1020. At this point, since John had his own computer, the A/W/A Team's game-writing turned a little more individualized instead of in group sessions. John primarily wrote Space Assailants and completely wrote Overflow using his own Atari 400, Eric authored Kooky Diver and Yahtzee, and Robert and Eric primarily wrote Piracy II. Robert and Eric also wrote a fairly sophisticated database program for use by the Eastern Michigan University Athletic Director, Gene Smith, who went on to become the A.D. at Ohio State University.

# PHREAKS AND PIRATES

When the A/W/A Team started at Eastern Michigan University in the fall of 1983, they met some classmates who also owned Atari computers. These new friends had hundreds of pirated Atari games on disks. At the time, Eric and Robert only had a cassette player, so their new college friends had to use a program to convert the disk games to cassette games. Most of them worked, but it was inconvenient for the friend and for Eric and Robert. Eric and Robert quickly decided to invest the $500 it cost to purchase an Atari 1050 floppy disk drive that held 92K bytes of storage on each side of a 5-¼ inch floppy disk. The floppy disks at that time only provided storage on the front side of the disk. To use the back side of a floppy, computer users had to punch a notch on the left side of the disk to remove the write-protection. Then the disk could be used on both sides, but even so it had to be manually flipped over to use the back side as storage. After Eric and Robert bought their Atari 1050 disk drive, it was a simple matter of them bringing a handful of blank floppy disks to their friends living in the EMU dorm rooms and coming home with all of the latest game releases.

At about the same time, Atari Bulletin Board Services (BBS's) opened for business and proliferated. These BBS's were like early web-sites prior to the wide-spread use of the Internet, complete with chat rooms, news, information, and file sharing. BBS's were accessed through a dial-up phone line, and had a public-facing area where all of the free information and chat rooms were accessed. Most BBS's also had a private password-protected area where pirated games could be accessed and downloaded. All it took was a little trust to be developed with the BBS system operator (sys-op) and you could normally get a password into the BBS's protected area and download even more games.

So in order to access these BBS's, Eric and Robert bought a 3rd party MPP 300 baud modem. The modem was blue and plugged into a joystick port, which on the Atari computer could be programmed as general I/O ports and provided the analog to digital conversion needed for the modem to work. This modem was used in conjunction with special Atari software that allowed the Atari to connect to the Eastern Michigan University mainframe and BBS's. Eric and Robert were not only able to download

hundreds of games, but they were some of only a handful of students who could actually program the DEC 10 and DEC 20 mainframe computers from the comfort of their own home.

One of the great Michigan bulletin board services was called the Country Peddler and was based in Whitmore Lake, Michigan. In those days, Whitmore Lake was a long-distance toll phone call from Ypsilanti. The Country Peddler was generous to its subscribers and put a relay phone in Ann Arbor, which essentially was a centralized phone hub for neighboring communities. Ann Arbor was a local phone call from Ypsilanti, and calling the Ann Arbor number forwarded the phone to the Whitmore Lake number. However, there were also many other great bulletin boards all over the country, and it would be too expensive to call them long distance and tie up the phone for hours downloading games.

So the next thing the A/W/A Team explored was phone phreaking. Phreaking was a way to make long distance calls for free. One way was to use a tone-generation program that could hijack a phone line and allow you to make free calls. There was a listing in one of the Atari computer magazines that provided touch-tone dial tones, including some tones not accessible on a normal phone that could presumably produce tones needed to call long distance for free. The A/W/A Team typed in the program, recorded the tones using a portable tape recorder, then went to a pay phone on Cross Street and attempted to play the tones necessary to get a free call. This was too complicated and required going to a pay phone which was nowhere near their Atari Computer, so this wasn't practical. The practical solution was to get free hacked long-distance codes from a bulletin board service. The way this worked was that you would call a free 800-number and then you would enter the long-distance code using a touch tone phone.

The problem was that Eric and Robert didn't have a touch tone phone. So they used the Atari program to make the touch-tone noises needed for the long distance codes to be entered by putting the phone next to the TV's speakers. This way, the A/W/A Team was able to access BBS's around the country through a free long distance phone call. Typically, after using the codes for a day or two they would stop working, but additional new codes were always available on the BBS's. Amazingly, these codes were only four digits long and in many cases sequential, so even without the codes being published on BBS systems, they could rather easily be guessed using trial and error. The ethical justification was that nobody (but the phone company) would be harmed by making these calls because once the owner of the code received a huge phone bill, they would easily be able to explain that they couldn't possibly have made all of those phone calls and were ultimately not billed.

# GAME OVER

The A/W/A Team broke up when Eric and Robert moved to Florida in 1987. By that time, the world had moved onto PC's and Macs, and nobody really programmed for the Atari 8-Bit machines anyway. Eric wrote one last game, Soko-Ban, in 1988. After that, the Atari 400, Atari 800XL, the printers, the modems, the peripherals, the disks, and cassettes went into the closet and attic, rarely to be used again.

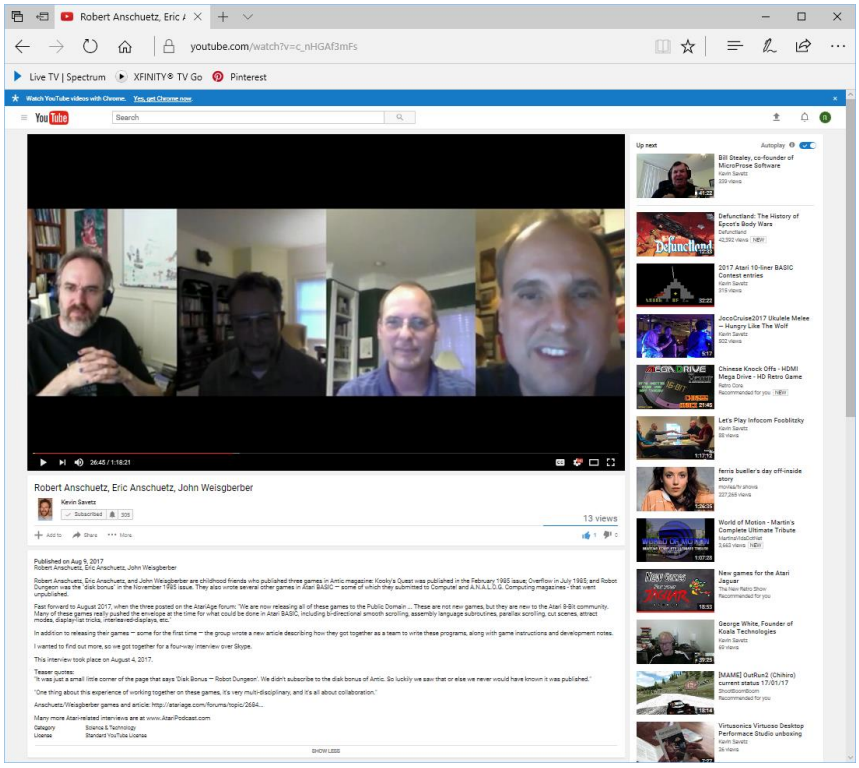
In about 1990, a device called SIO2PC interface came out that allowed that Atari to connect to a PC. This was handy because all of the floppy disk games could be copied onto a hard-drive on the PC and

persevered. Around that same time, several Atari 8-Bit emulators started to become available to allow playing Atari 8-Bit games on a PC, and almost all Atari commercial games were distributed on USENET COMP.SYS.ATARI.8BIT servers. Using SIO2PC, The A/W/A Team backed up most of their BASIC game onto "virtual" floppy disks that could be used by the emulators. After the games were copied to a PC and emulators could run them, there really wasn't a need to pull the original computers out of the closet and attic.

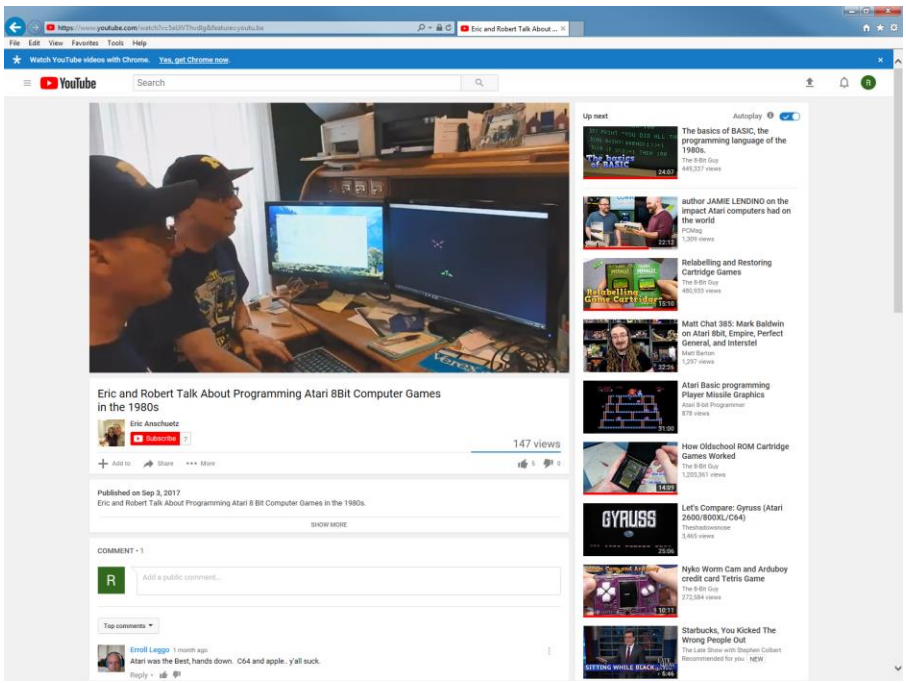
# LASTING LEGACY

The A/W/A Team is credited on the list of "Giant List of Classic Video Game Programmers" http://dadgum.com/giantlist), which documents most of the classic era (1980's) of videogame programmers. The articles and BASIC code for the three Antic-published A/W/A games (Kooky's Quest, Overflow, and Robot Dungeon) can be searched for on the Internet, and will always be available for download. However, there were several other great games written by the A/W/A Team that should be documented somewhere, so this document is the place to do it. That includes the two games sold to COMPUTE! (Kooky Klimber and Night Rescue), plus several others that were never published. Unfortunately, there were several other games on cassettes and floppy disks that were written by the A/W/A Team that weren't backed up and are probably are lost forever, including the phoenix game, the pyramid game, the racing game, the space game, and EMU Athletic Department database program.

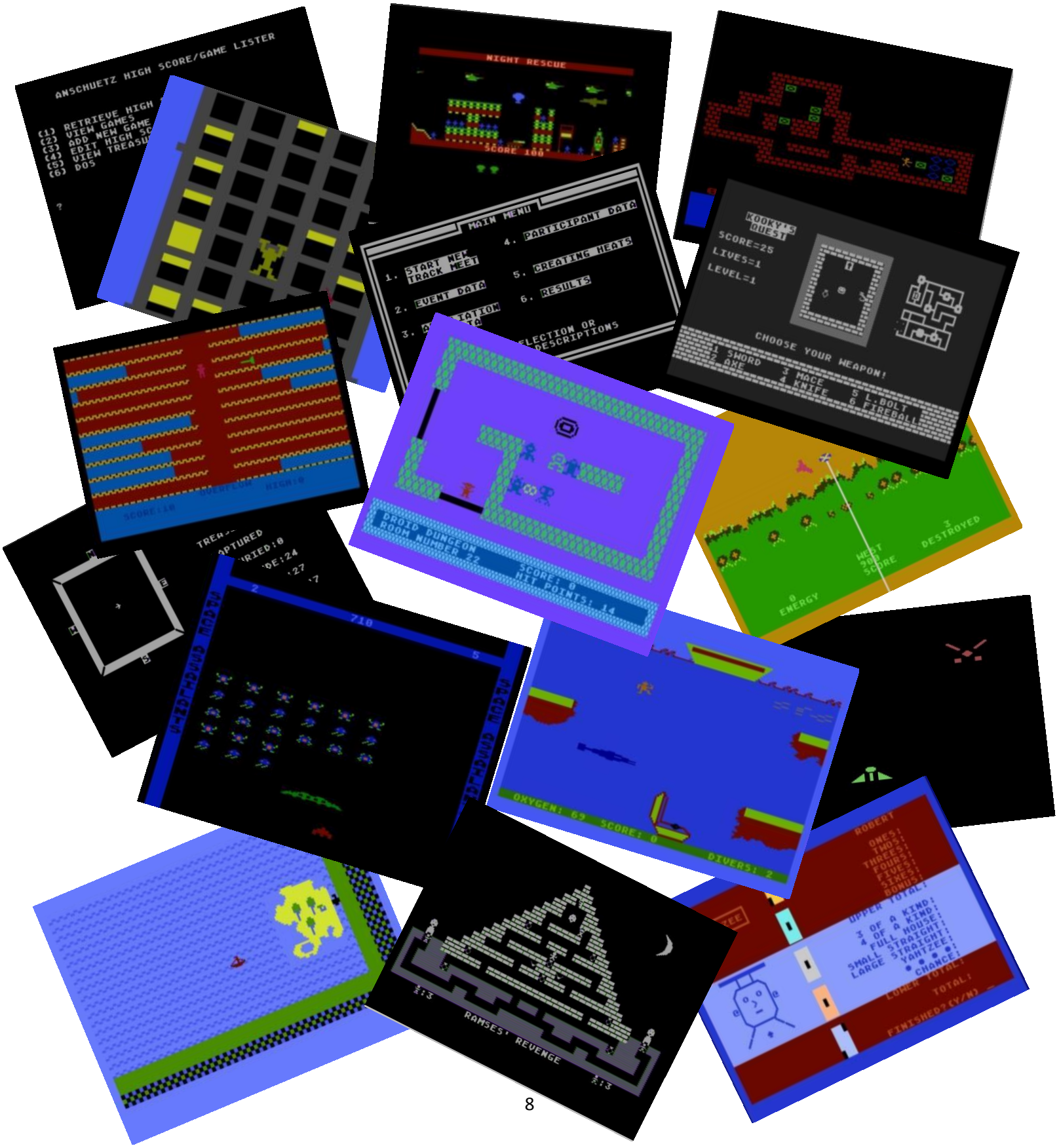
The good news is that many of the games are documented here and are still playable through emulation. All of the games have now been released and posted to the Atari 8-bit games repository http://www.atarimania.com. Links to each of the specific games for download have been provided below in the section discussing each game. On August 4, 2017, the A/W/A Team was asked to participate in an interview for the Antic Atari Podcast with Kevin Savetz. The audio for the podcast interview is at this link: http://ataripodcast.libsyn.com/antic-interview-297-robert-anschuetz-eric-anschuetz-john-weisgberber-antic-magazine-games. The YouTube video for this podcast interview is at this link: https://www.youtube.com/watch?v=c_nHGAf3mFs.




On September 2, 2018, Robert and Eric played each and every one of the games that they had written and recorded their reactions to the games and their recollections making them. The YouTube video for this podcast interview is at this link: [https://www.youtube.com/watch?v=5eUiVThvdlg](https://www.youtube.com/watch?v=5eUiVThvdlg).



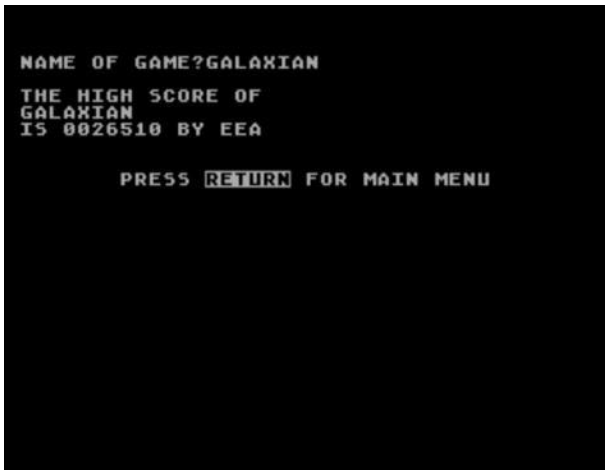
# A/W/A BASIC GAMES



# ANSCHUETZ HIGH SCORE / GAME LISTER (1985)

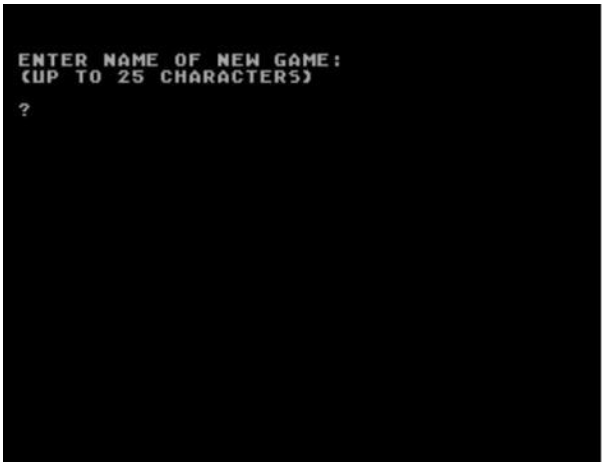
http://www.atarimania.com/utility-atari-400-800-xl-xe-high-scores_31444.html

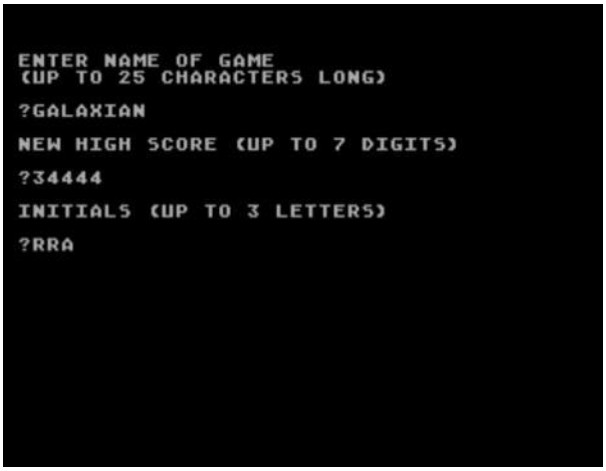


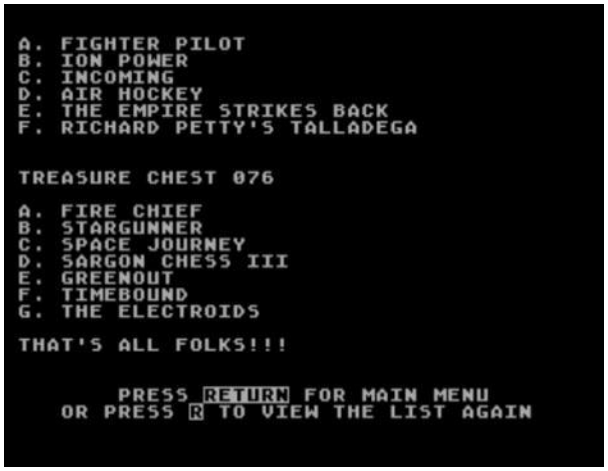










**Game Play**

This high-score database program was created to keep track of the A/W/A Team's game disks (called Treasure Chests, numbered from 1-76) and their high scores. The program allows the user to retrieve a specific game's high-score, view a listing of all of the games, add a new game to the list, edit an individual high-score, and view a list of the contents of the Treasure Chest game disks. The option to show a listing of the high-scores includes an option to alphabetize the list.

**Recollections**

Microsoft Access and Excel didn't exist in the 1980's, so the A/W/A Team wrote their own database program and accompanying file format to keep track of all of their game disks and high-score data. The program is relatively straight-forward and is mostly disk read/write routines to store the data. The routine to alphabetize the game list is actually an assembly language sort routine. The A/W/A Team found this sort routine in a magazine listing and used it in the program. The methods used in writing database information to the disk drive was used as a foundation for the EMU Athletic Director's database program. Eric and Robert encouraged all of their friends and relatives to have at least one entry in the high-score database. They would often find an obscure game and then encourage a friend to play it to be immortalized in the high-score database. Prophetically, 35 years later, these same initials can be found throughout the high-score listings, many of which the A/W/A Team is now unable to remember who they belonged to!

# ALIEN ASSAULT (1983)

http://www.atarimania.com/game-atari-400-800-xl-xe-alien-assault_31443.html




**Game Play**

Alien Assault is clearly influenced by Atari Missile Command. The object of the game is similar - missiles rain down from space onto satellites that must be protected by a laser fired by a central base. You start the game with 35 protective lasers, as indicated by the “Energy” counter at the bottom of the screen. The "Destroyed" counter shows how many of the satellites have been destroyed. Unlike Missile Command, only one missile comes down at a time, but the screen can be scrolled “West” and “East” (left and right) to observe the missile and shoot it down. When you shoot down all of the missiles in a set, another set appears with a different colored background, which is another homage to Missile Command. Once all of the bases are destroyed or you run out of energy, your game ends.

The following description of the gameplay was written by the A/W/A Team to be submitted to a computer magazine along with the game.

*You have been stationed on the moon in the year 2136. The reason? To protect solar collectors from possible destruction. You will do your protecting from within a super-advanced, armed battle tank. The tank is made of a very strong alloy that cannot be destroyed. Think you are up to the task?*

*The moon was discovered to have a vast resource of area available that is ideal for the positioning of solar collectors. These collectors provide much of the earth's energy and are mandatory for the survival of humanity. Your mission: protect the moon, and more specifically, the solar collectors from the ongoing ALIEN ASSAULT.*

*You begin your mission in the eeriness of night. Intelligent radar indicates that you will have to continue your frenzious attack on the alien bombs on through the spectacular lunar sunrise, onto midday, through the next sunset, and onto another day. Before you begin your mission, you received a supply of 50 units of energy. As you shoot at the alien bombs with your lasers, however, your supply of energy will rapidly begin to decrease at the rate of one unit per each laser fired.*

*Earth intelligence has revealed that on the first series of attacks, the aliens will send out ten missiles aimed to destroy ten of the solar collectors. During each successive onslaught, five additional missiles will be sent until the maximum total of 40 is reached. In addition to more enemy bombs to worry about, on every other wave of bombings, the amount of solar collectors that the aliens have to destroy is decremented by one. This is due to the fact that the solar technicians are too busy fleeing back to earth to have time to make the necessary repairs. Malfunctions do happen occasionally even in this advanced age.*

*As each alien missile is launched by the alien armada toward the solar collectors, the indicator in the lower middle of the display will indicate whether the missile is in the west (left) or to the east (right) of your tank. To shoot a laser at one of the alien missiles, position your crosshairs on top of the missile and press the fire button. A laser will be sent from your tank toward the spot in space directly in line with where the crosshairs are aimed.*

*If you manage to line the crosshairs directly on top of the alien missile, the missile will be destroyed. The number of missiles remaining to be sent will be decremented by one. If you should miss the missile, one of two things may happen. First, the missile may hit a solar collector. If this should happen more than the amount of times allowable for each ambush, your game will be over. Second, the missile may crash harmlessly on the surface of the moon, in which case nothing happens. At the end of each set of ambushes, the color of the*

*background will change to indicate the rising of setting of the sun. Every few sets, the missiles will speed a bit in their descent.*

*ALIEN ASSAULT is a difficult game which requires manual dexterity as well as a strategy. Here are some hints. Always keep an eye on the indicator in the middle of the screen. This will tell you which direction to begin moving in. Always begin moving immediately. This will help you get to the enemy faster.*

*Keep a mental note of where you are in relationship to the entire playing field. If you find you cannot travel any more in the direction in which you wish, you have reached the outside boundaries and should change directions immediately. As you become more advanced in ALIEN ASSAULT, you will have a feel for when you have begun to travel too far in a particular direction. Sacrifice a solar collector if you feel you are drifting a long way away from the center of the playfield. This may mean many collectors saved in the long run.*

*Keep an eye out for the energy remaining. This means take carefully placed shots instead of machine-gun style. If you do shoot machine-gun style, you will find that your energy supply will quickly run out and you will be forced to watch helplessly as your last remaining solar collectors are destroyed.*

*Also watch the number of collectors destroyed in the lower right-hand corner of the screen. Always compare this to the number of collectors that are permitted to be destroyed in the particular ambush. Remember, this number starts at ten and is decreased by one on every other wave.*

*One final suggestion is to always keep your crosshairs positioned near the bottom. This will give you that all-important last shot. This may prove to be the difference in lasting another ambush or not.*

**Recollections**

Alien Assault was one of the early A/W/A games written before the team had a disk drive. How do we know that? Because the code overwrites the area of the disk drive handler, and therefore doesn't work if you run it off the disk! It wasn't until 2017 that Eric Anschuetz discovered this major flaw, and re-wrote the BASIC code so that it runs in a protected area of memory and can now run on disk some 30 years later! The BASIC code listings show both the original code (circa 1983) and the updated code (2017).

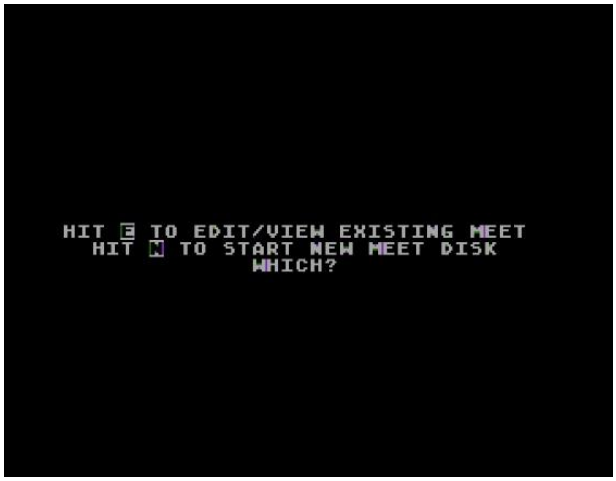
The missile coming down from the screen is a Player/Missile graphic, as is the target indicator controlled by the joystick. The laser that is fired at the satellite is another Player/Missile graphic. This game makes use of parallax scrolling, which means that the satellites in the foreground scroll at a faster rate than the mountains in the background. This mirrors the effect in Williams Moon Patrol, and may have influenced the game a bit.

More information about the technical aspects of this game can be read in the original accompanying article sent to computer magazines, A/W/A Game Write-Ups section of this document.

# EMU TRACK MEET DATABASE (1987)

http://www.atarimania.com/utility-atari-400-800-xl-xe-emu-track-meet-database_31477.html



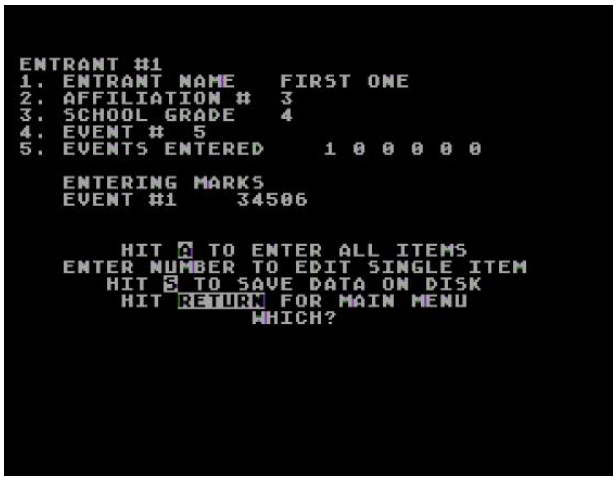


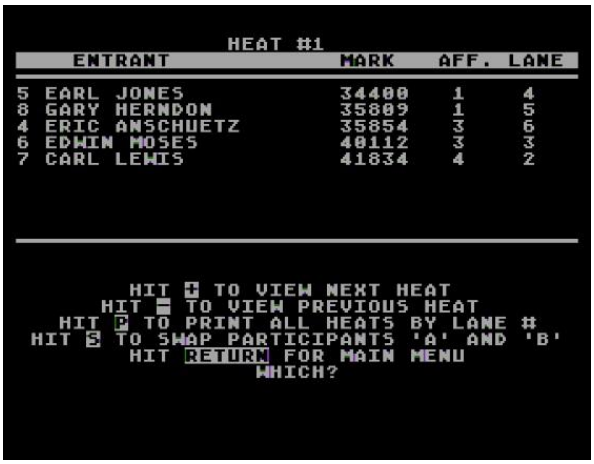


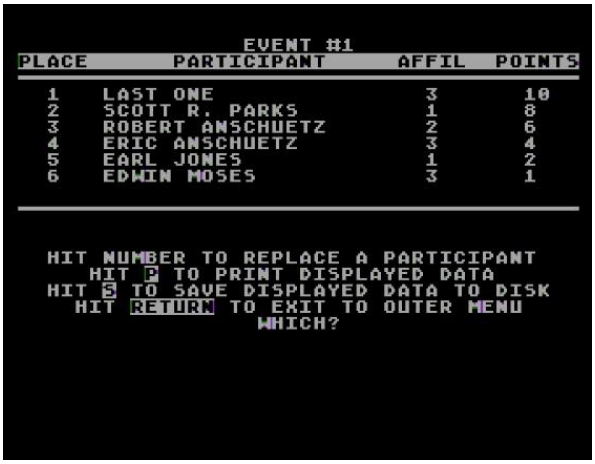









HEAT #1
ENTRANT MARK AFF. LANE
5 EARL JONES 34400 1 4
8 GARY HERNDON 35809 1 5
4 ERIC ANSCHUETZ 35854 3 6
6 EDWIN MOSES 40112 3 3
7 CARL LEWIS 41834 4 2
HIT + TO VIEW NEXT HEAT
HIT - TO VIEW PREVIOUS HEAT
HIT P TO PRINT ALL HEATS BY LANE #
HIT S TO SWAP PARTICIPANTS 'A' AND 'B'
HIT RETURN FOR MAIN MENU
WHICH?


EVENT #1
PLACE PARTICIPANT AFFIL POINTS
1 LAST ONE 3 10
2 SCOTT R. PARKS 1 8
3 ROBERT ANSCHUETZ 2 6
4 ERIC ANSCHUETZ 3 4
5 EARL JONES 1 2
6 EDWIN MOSES 3 1
HIT NUMBER TO REPLACE A PARTICIPANT
HIT P TO PRINT DISPLAYED DATA
HIT S TO SAVE DISPLAYED DATA TO DISK
HIT RETURN TO EXIT TO OUTER MENU
WHICH?


DESCRIPTIONS
START NEW TRACK MEET
ENTER GENERAL DATA FOR NEW MEET
REVISE DATA ON EXISTING MEET
PRESS N FOR NEXT
PRESS M FOR MAIN
a/w/a


DESCRIPTIONS
EVENT DATA
ENTER/REVISE EVENTS
PRINT OR LIST EVENT
PRESS N FOR NEXT
PRESS M FOR MAIN
a/w/a




**Game Play**

This program was written to keep track of various administrative data for the EMU Athletic Department. The software was composed of a main program that launched 6 other programs based on user input. The user could manipulate data based on the following items, each corresponding to a different program: 1. Start New Track Meet, 2. Event Data, 3. Affiliation Data, 4. Participant Data, 5. Creating Heats, and 6. Results.

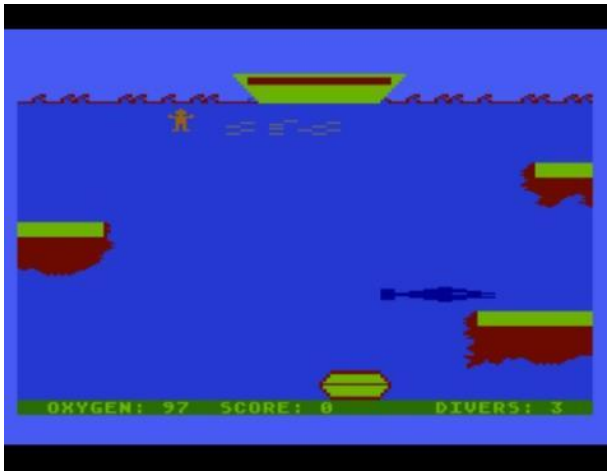
**Recollections**

The A/W/A Team was introduced to Mr. Smith by their friend Scott Parks, whose dad was the Men's track and cross-country coach at EMU. Mr. Smith had knowledge of a very expensive database program tailored for Athletic Directors, and wondered if EMU students could replicate the program. The A/W/A Team met with the A.D. and interviewed him about the functionality required of the program. The A/W/A Team quickly developed a prototype implementing some of the functionality and showed it to the A.D. However, it really wasn't a polished program and ran exclusively on the Atari computer instead of on the EMU mainframe computer. Interest died after that, and the program was never completed or delivered.

# KOOKY DIVER (1986)

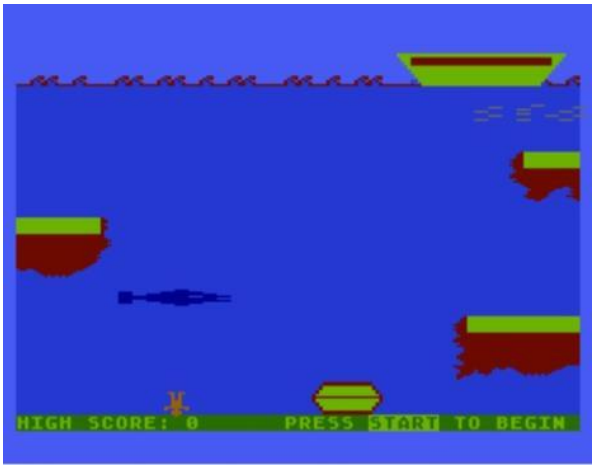
http://www.atarimania.com/game-atari-400-800-xl-xe-kooky-diver_31438.html










**Game Play**

The object of Kooky Diver is to swim from your boat to the ocean floor, retrieve treasures from a giant clam, and swim back to return it to your boat. All the while, you must avoid a school of fish, a shark, and the closing clamshell. You only have three divers to complete your mission. While not moving, you are constantly descending at a slow rate toward the ocean floor. You can use your joystick to move in any direction to move and avoid the shark and fish. If you are hit by a shark or run over by the school of fish, you die immediately and precede head-first toward death and Davy Jones' Locker. When you reach the clam, you must wait until it opens up before you can retrieve the treasure. The treasures change each time you collect one. The first treasure is a pearl, the second one is a trophy cup, and so on. The opening and closing of the clam is random, so you must tread water waiting for it to open, all while your oxygen is running out. Once the clam is open, the diver must swim to the treasure, press the fire button to retrieve the treasure, then quickly make it back to the boat before the oxygen level reaches zero. The oxygen runs out much faster when you are weighted down by the treasure. Once you return the treasure, your mission starts over and you must retrieve the next treasure until all three of your

divers have died. You can place the treasures on the coral reefs to each side as a way to not run out of oxygen. Some of the larger items require this "rest stop" in order to bring the item to the surface. If your oxygen is less than 50 percent when you reach the surface, it will be restored to 100 percent.

The following description of the gameplay was written by the A/W/A Team to be submitted to a computer magazine along with the game.

> *Yes, he's back and he's kookier than ever. But this time, instead of climbing another building, Kooky has taken the opposite extreme and has begun a search for lost treasure. On his dives, he will encounter black pearls, treasure chests, pistols, vases, steering wheels, and many other treasures. But what Kooky doesn't know as he begins his expedition is that not only are there a dangerous shark and a school of man-eating piranha to block his path, but that a giant clam has swallowed all of the treasure! You, as Kooky, will have to dive for the various treasures and attempt to bring them up to the salvage boat. To move Kooky in any particular direction, just move the joystick in the corresponding direction. To tread water, push the joystick in any of the diagonal directions. You must time it just right so that the clam's mouth is open when you touch the treasure in order to have a chance to make a successful salvage. If the mouth is open and you are touching the treasure, merely press the fire button on the joystick and you will have picked up the treasure. Of course, if you touch any part of the clam when its mouth is closed, you will lose a life. After you have a treasure in your possession, you will use up your limited supply of oxygen rather quickly. Luckily, there are three coral reefs for you to deposit your treasure on while surfacing for more oxygen. Provided that your current oxygen level is below 50 units, you will completely fill your tanks every time you bring your head out of water. To drop off a treasure on any reef, all you must do is make sure that Kooky is only touching the yellow portion of the reef. At this point, the treasure will automatically be set there with no further actions required. To pick up a treasure again, simply press the fire button while touching the treasure as before. Of course, any collision with the piranha or the shark will cause instant death and send Kooky plummeting toward the ocean floor. After three deaths, your game will end and the high score will be shown. At this point, you can press the "START" button on the computer to start a new game with three new divers. You may end up doing this quite a few times at first because after all, nobody ever said that diving was easy!*
>
> *The ultimate goal of the game, however, is to deposit your treasure in the boat that is moving across the top of the screen. To do this, just touch the boat and your treasure will appear inside. At this point, a new treasure will appear inside the clam, and you can now try to get this treasure to the boat.*

**Recollections**

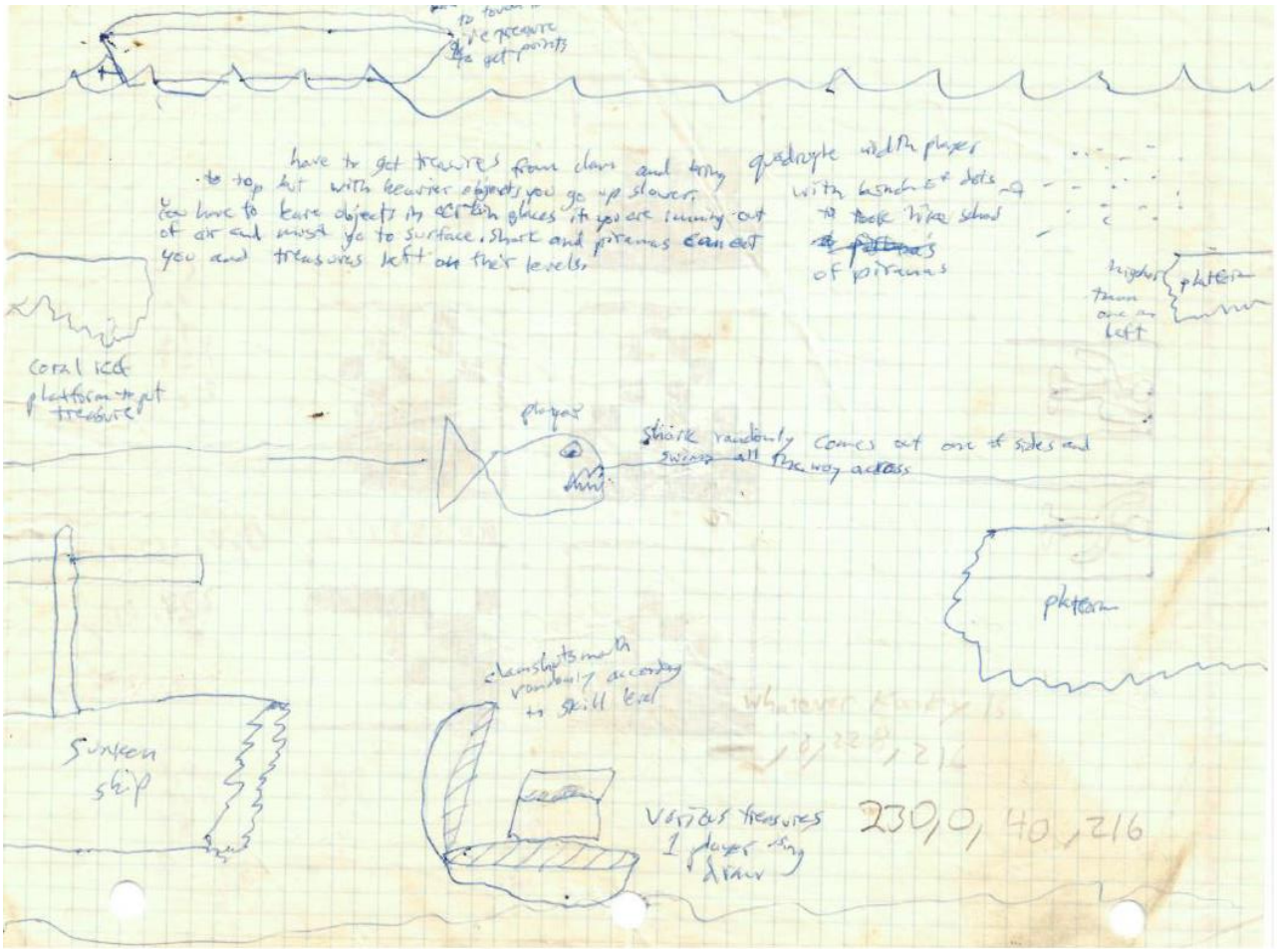
Kooky Diver was primarily written by Eric Anschuetz. The game is really unique and not patterned after any commercial arcade or home game. The top row with the waves and the boat implement a rough scroll technique. The diver, fish, shark, and pearl are obvious uses of the four Player/Missile graphics allocated on the Atari 8-bit computer. The coral reefs, the waves, the boat, and the clam shell are implemented using redefined characters.

A technique that is used in this game and many other A/W/A games is the background sound effect, since games are much more boring without them. Since the games were written in BASIC, there really wasn't time to play a real melody during the display list interrupts. So the A/W/A Team used a technique where an assonant tone is produced through the Pokey sound chip and it sounds as if it is a wavering tone. It's kind of annoying after a while, but it is much better than silence and could be accomplished with one line of code! There was a lot of experimentation with the various parameters of the BASIC SOUND command to get a semi-pleasant-sounding background tone that sounded right.

This game is really polished and doesn't have any glitches or bugs present as in some of the other games due to the fact that this game doesn't really push the envelope of what can be done with Atari BASIC. The game initializes really quickly and it doesn't take long to start a new game at the end of a game.

A really good professional feature of this game is that at the end of the game, everything doesn't just freeze on the screen or go back to the start menu. After the game is over, the fish and shark keep swimming, and the bottom display toggles between the score of the last game and the high score / restart prompt. This implementation of an "attract mode" was something almost non-existent in type-in home games.

This game was programmed in one long day after Eric dreamed of the game the night before. The concept art that Eric drew the morning after his dream looks nearly identical to the finished game. The game was quickly coded from there. From concept to game in 24 hours was pretty amazing for a 20-year-old!

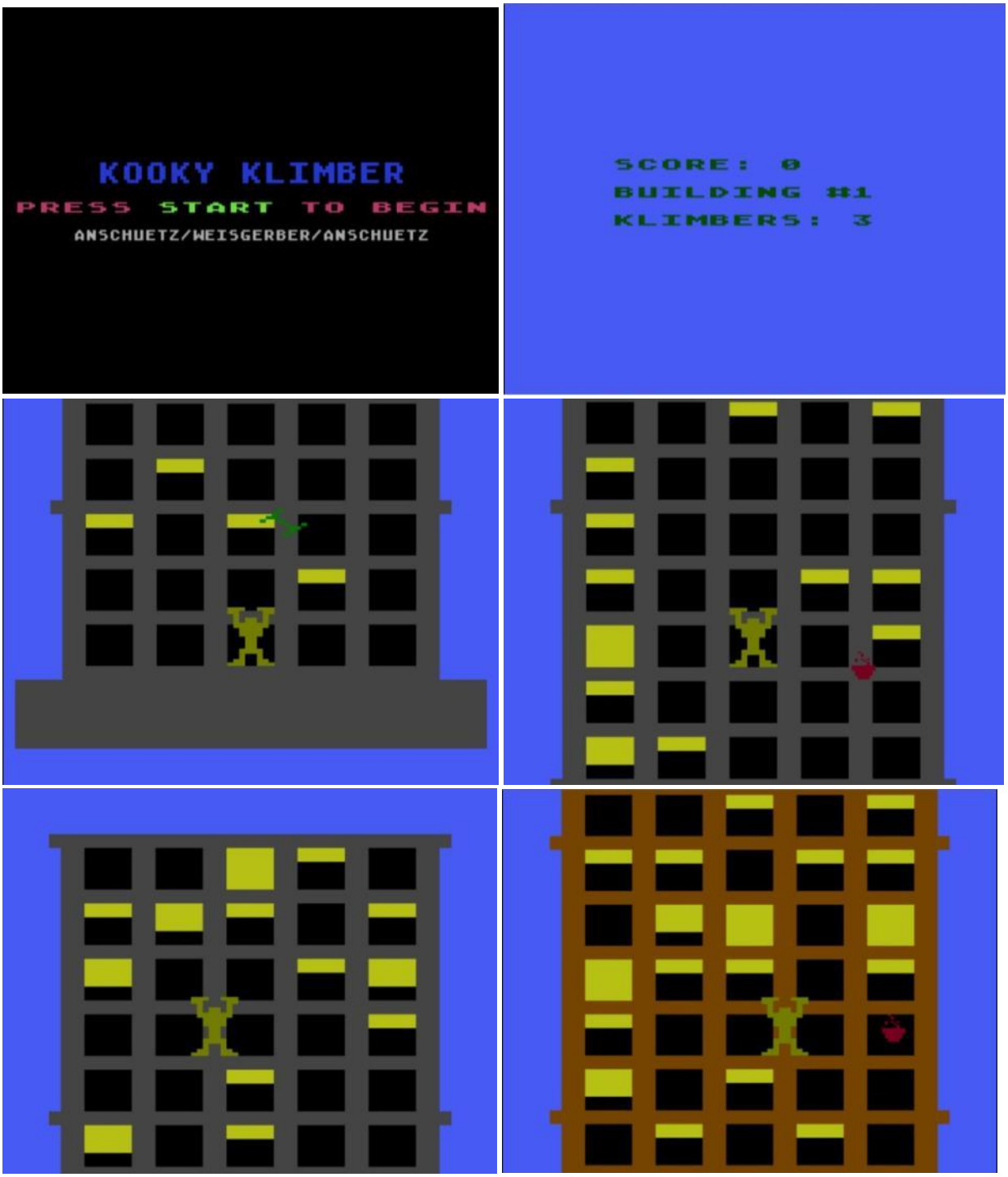



Eric used his first display list interrupt in this game – an advanced technique that allowed some code to be written in the brief time that the TV finished scanning one line and then waited to scan the next line (all at 60 hertz). The DLI was used to change the color of the score line at the bottom to green, thus using an additional color not normally available in the Atari's four-color display palate.

More information about the technical aspects of this game can be read in the original accompanying article sent to computer magazines, A/W/A Game Write-Ups section of this document.

# KOOKY KLIMBER (1984)

http://www.atarimania.com/game-atari-400-800-xl-xe-kooky-klimber_31437.html




**Game Play**

The gameplay for Kooky Klimber is simple – reach the top of the building by avoiding getting your hands caught in closing windows, while also being careful not to get hit by dropped dumbbells and flower pots. To move up a flight of the building, you must pull back on the joystick and then push forward. You can move left and right to dodge falling objects and closing windows by pushing left and right on the joystick. Once you near the top of the building, the objects quit falling because there are no windows above you from which they would fall. Once you reach the top of the building, you start all over at the bottom of the next building, represented in a different color. You continue the game until all of your three Kooky Klimbers are knocked off the building.

The following description of the gameplay was written by the A/W/A Team to be submitted to a computer magazine along with the game.

> *In this game, you are Kooky Klimber. You have an incurable urge to climb tall buildings. After a period of initialization, you find yourself poised at the bottom of the first (grey) building. To ascend the building you must pull the joystick back and then push it forward. After this, your klimber will automatically climb to the next story. To move left and right simply push the joystick in the corresponding direction. Sound easy? Well, as you will soon discover, it's not!*
>
> *In the course of your climb, the people inhabiting the building will be opening and closing the windows. If one of these windows should close on your hands, you will find yourself falling head first towards the bottom of the building. A second nemesis directed towards your klimber is the falling objects. Clumsy residents above have a deadly habit of knocking flowerpots off of their windowsills. These flowerpots can do lethal harm to your klimber. If they should touch any part of Kooky Klimber, he will lose his grip on the building and fall to a quick death.*
>
> *Another clutsy group of people above you are the construction workers. They are continually dropping steel girders which flip through the air while they fall. As in the case with the flowerpots, any contact between your klimber and the girder will cause you to slip and fall.*
>
> *Your game starts with three Kooky Klimbers. A bonus klimber is earned after achieving 10,000 points. Once you reach the top of any building, you will be transported to the bottom of another building with a different color and harder skill level.*

**Recollections**

Kooky Klimber was based on the Nichibutsu arcade game called Crazy Climber. The A/W/A Team played Crazy Climber at Mickey Rats arcade in downtown Ann Arbor, and loved the daredevil theme of climbing up a building. Crazy Climber featured comic-book-style cartoonish characters, catchy background music, and it was one of the first games that used a digitized voice ("Go For It!"). After

reading about vertical scrolling and Display List modification techniques for the Atari computers, the A/W/A decided that this would be the ideal game to copy. Many "ripoff" clones in the 1980's used titles very similar to the original game, but with enough differences to presumably avoid lawsuits. For their Crazy Climber knockoff, the A/W/A Team thought of the clever title of Kooky Klimber which also a "double K" alliteration. The game makes use of the Player/Missile graphics available in the Atari. The title character was one obvious Player/Missile graphic, then there were other Player/Missile Graphics in the form of items that fell from above, including a flower pot and a spinning dumbbell, all patterned after the original game.

The main player was animated so that he moved his arms and legs up and down as he climbed up the building. These animations were fairly easy to sketch out in the 8x8 character limitation. The A/W/A Team used graph paper to draw the characters, then drew the 8-bit binary multipliers in the columns above and wrote the 0-255 value corresponding to each row of the character on the right side of the character. Each of the 4 frames in the animation was drawn out, and repeatedly re-written in the Player 0 character map as the animation sequence took place with each move up the building. The body of the character "scrunched", or "compressed," as the character animated, and this was actually an unintended bug due to incorrect positioning of the animations and a missing a row of the data when it was programmed. Since it actually looked pretty good, the A/W/A Team kept it in the animation sequence. There was also a nifty 4-note sound effect that corresponded with the animation sequence with each move up the building.

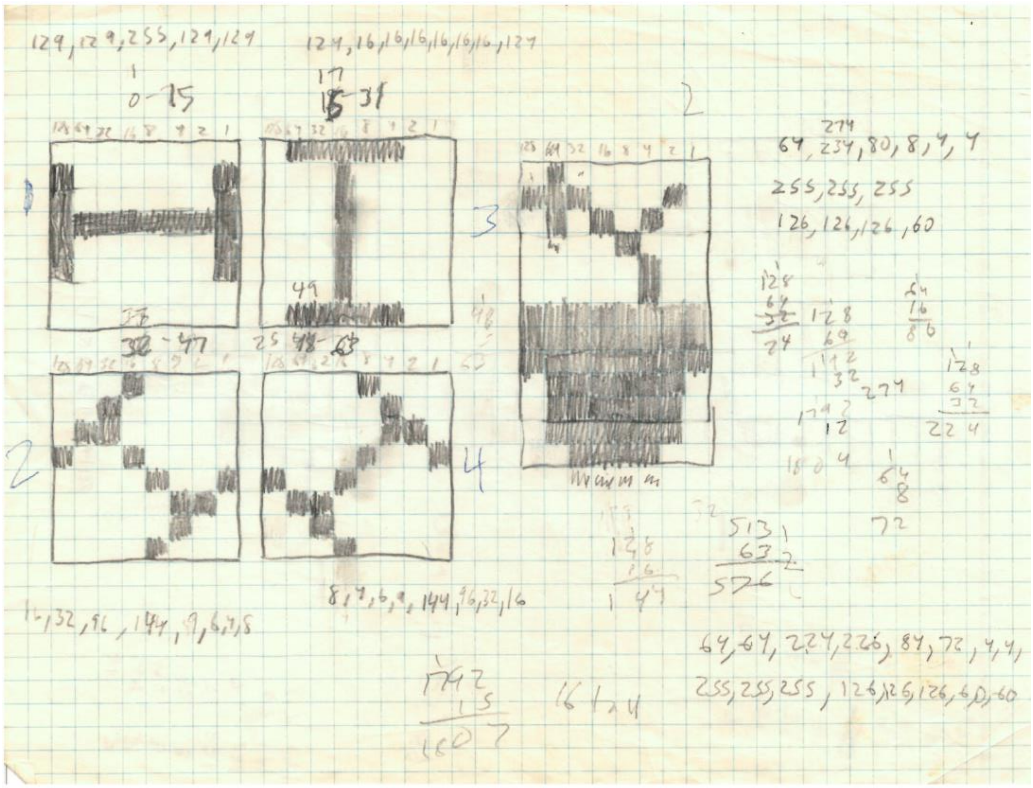
Each level had opening and closing windows that were modified in the Atari's Display List. The timing of the windows opening and closing was perfect for gameplay, and therefore made a really challenging game. When the windows closed on the hands of Kooky Klimber, or he was hit by a flower pot or dumbbell, he fell to the bottom of the building. The scrolling for this effect was really quick, and often Display List glitches could be seen because Atari BASIC couldn't handle the modifications to the memory fast enough. The glitch was something like a quick jerking of the screen, that would only last $1/60^{th}$ of a second, but it could be seen occasionally. The vertical scrolling of the building was implemented as rough scrolling (8 bits at a time) as opposed to fine scrolling (1 bit at a time). The glitch occurred when the graphics scrolled though a "page" of memory (256 bytes). To modify the Display List to do this required changing two bytes (most significant byte (MSB) and least significant byte (LSB)) on the 8-bit computer to address the 65536 (2^16) memory space. These changes were supposed to be done during the vertical blank time when there was just enough time to write a little bit of code during the vertical blank interrupt (VBI). When done in BASIC outside of the VBI, each byte (MSB and LSB) needed to be modified sequentially and that caused the glitch because for a brief period, the memory location being pointed to was not correct.

Once Kooky Klimber reaches the top of the building, the score is displayed on a separate screen, and he starts at the bottom of the next level. The next building is implemented as a different color, the windows open and close at a faster rate, and the items fall from the top of the screen faster as well. There really wasn't room to display the score and lives remaining on the main scrolling screen, so it was only displayed at the end of each building and at the end of the game. Another problem was that it was

difficult to mix text and graphics, so a display list modification would have been necessary to position a line of text, and that would have been difficult to implement.

The falling flower pots and dumbbells were very well done. The dumbbells rotated as they tumbled from the sky. Also, with each vertical move of Kooky Klimber, the falling objects moved in tandem with the building. In other words, instead of falling at a rate of speed independent of the moving building, as the building scrolled down, the falling objects moved quicker in relation to the sky so they would move with building at the same rate. This extra level of realism took extra time to implement, and many arcade games even ignore these kinds of details. Every once in a while, the falling objects had a glitch where they were distorted. Obviously, it was some sort of memory over-write that may have occurred when Kooky Klimber was loaded after another program, but the bug was never fixed. It was rare, but occasionally a misshapen falling object comes on the screen.

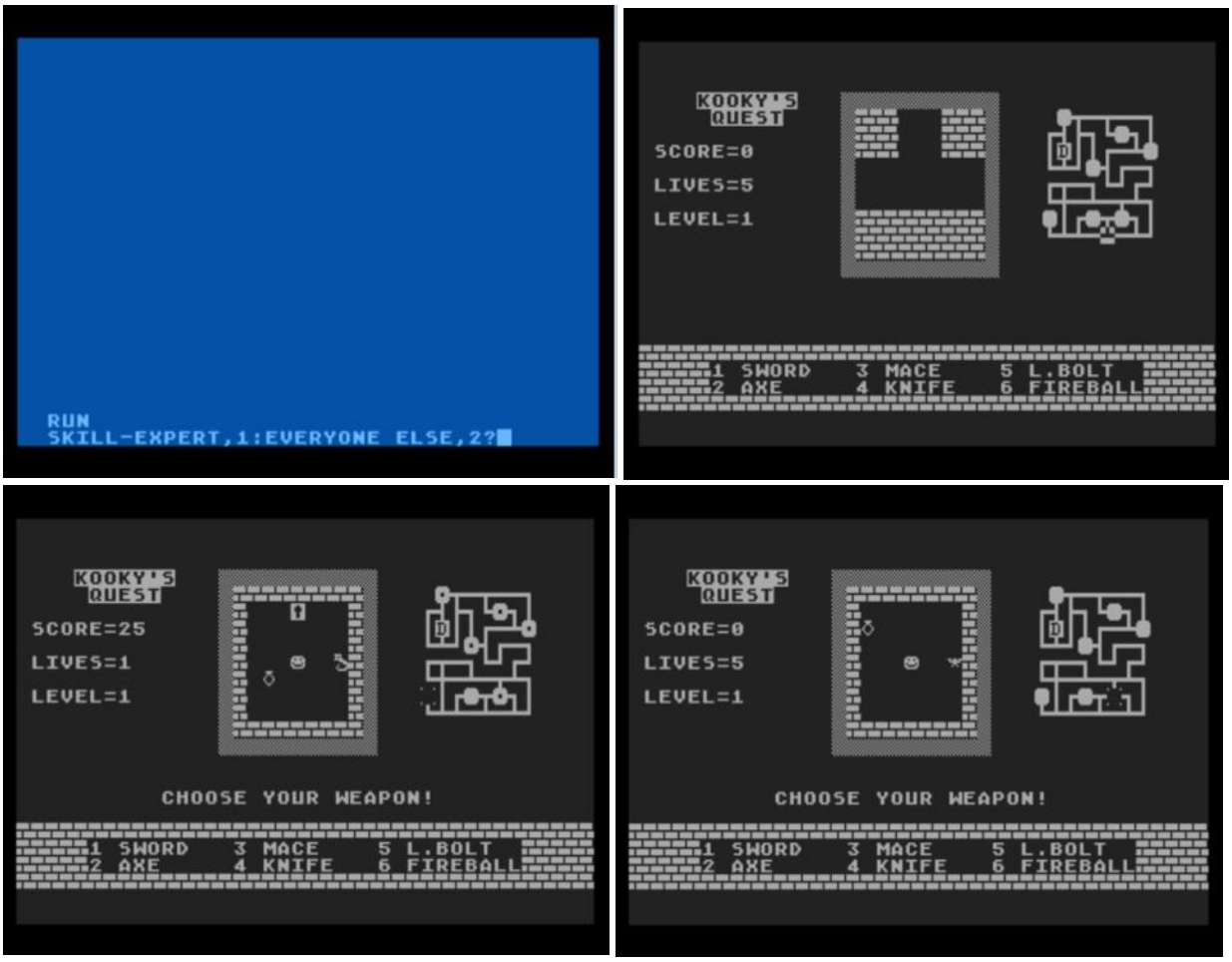
Kooky Klimber was the first game that was sold by the A/W/A Team. They sent a long technical write-up describing the game's instructions and programming techniques along with a cassette of the BASIC listing to COMPUTE! Magazine. A few weeks later, they received a letter from the editor offering $400 for the rights to publish the games. The A/W/A Team was absolutely thrilled that they got paid for a game they loved writing and playing initially for only themselves. Unfortunately, the game was never published by COMPUTE!



More information about the technical aspects of this game can be read in the original accompanying article sent to computer magazines, A/W/A Game Write-Ups section of this document.

# KOOKY'S QUEST (1985)

http://www.atarimania.com/game-atari-400-800-xl-xe-kooky-s-quest_2834.html




**Game Play**

The following description of the gameplay was written by the A/W/A Team to be submitted to a computer magazine along with the game.

> *Kooky's Quest is an action adventure game. You play using the keyboard and a joystick in controller jack one. There are two skill levels: expert and beginner. In a beginner game, you have a map of the dungeon to follow. If you like the challenge of mapping a game, you should play expert, as you can only see the section of the dungeon you are occupying.*
>
> *Trapped in an ancient dungeon, your only hope lies in descending to the lowest level and retrieving the magic key. By returning to the first level with the key*

*and placing it in the lock located in one of the level-one rooms, you can free yourself from the evil dungeon and win the game. You have ten lives with which to accomplish this task.*

*In order to move from one level to another, you must obtain a certain number of treasures (rings and chests). You need only one treasure to go from the first level to the second, but you need five in all other cases. Once you have enough treasures to change levels, you simply move to the up or down staircase, designated by a U and a D on the map. If you are playing without the map, you can only identify a staircase by the sound effects for going up and down, because the stairs aren't visible in the view of your immediate vicinity.*

*Getting treasures is not an easy task. In each room, you will find a treasure and a monster. It is important to get to the treasure before the monster gets it, before you kill the monster, or before the monster kills you, since, in any one of these cases, that treasure will be lost to you. If you clear all the rooms on any level and don't have enough treasures to move to another level, you will be forced to wander that level forever.*

*As you enter a room, you are asked to choose which weapon you will use. Listed at the bottom of the screen are the six weapons available to you. Simply press the number of the weapon you want to use. Three of the weapons, the sword, axe, and mace, are swinging weapons. To swing any one of these, you press the fire button, then push the joystick to the left or right. The knife, lightning bolt, and fireball are throwing weapons. The procedure for using one of these is the same as for a swinging weapon, but it will travel across the room until it hits a monster, a treasure, or the wall. Your choice of weapon is very important, since each type of weapon has a different effect on each monster, taking from one to six hits to kill the monster.*

*There are six varieties of monsters you may encounter during your quest. These include vampire bats, giant hobgoblins, wicked skeletons, venomous serpents, decaying mummies, and worst of all, malevolent spiders which can catch you in their webs and inhabit only the lowest level.*

*As a variation of Kooky's Quest, instead of attempting to escape the dungeon with as many lives as possible, you can go for points, seeing how many you can accumulate before you lose all your lives. A treasure is worth five points and a monster is worth ten.*

The following gameplay instruction for Kooky's Quest was published in Antic Magazine.

*Trapped in an evil dungeon, your only hope of escaping is to descend to the lowest level, retrieve the magic key, and use it to unlock the portal at the*

*uppermost level. Playing Kooky's Quest uses the keyboard and a joystick in port one. There are two skill levels. In the beginner game, you're aided by a displayed map of the dungeon, which is erased for the expert game. To move from one dungeon level to another, you must obtain enough treasures-rings and chests. You need only one treasure to descend from the first level to the second, but you'll need five to reach each of the deepest dungeons. Once you have enough treasures to change levels, go to the up or down staircase, shown with a D or U on the map. If you are playing without the map, you can identify a staircase by sound effects, because the stairs aren't in your view.*

*Getting treasures is not an easy task. In each room, you will find a treasure and a monster. It is important to get to the treasure before the monster gets it, before you kill the monster or the monster kills you. In each case, you'll lose the treasure. If you clear all the rooms on any level and don't have enough treasure, you'll be forced to wander that level forever-or until you stop the game. You have ten lives for the entire game.*

*You'll be asked to choose a weapon as you enter a room. Available weapons are listed at the bottom of the screen and can be selected with numbers. The axe, mace and sword are swinging weapons-press the fire button and move the joystick from left or right. The knife, lightning bolt and fireball are throwing weapons. They are used in the same way as swinging weapons - except they continue moving in a straight line until they hit something. Choose your weapons with care. Each has different properties. You'll find that some require six hits to kill, others only one hit. Note that while fighting, you can only attack to the left or right, and can't protect yourself from above or below.*

*You can encounter six types of monsters - vampire bats, giant hobgoblins, wicked skeletons, venomous serpents, decaying mummies and malevolent spiders that inhabit only the lowest level. Your adventurer, Kooky is shown as a happy face symbol. Instead of escaping the dungeon, you may choose to accumulate points. Treasures are worth five points, defeated monsters ten.*

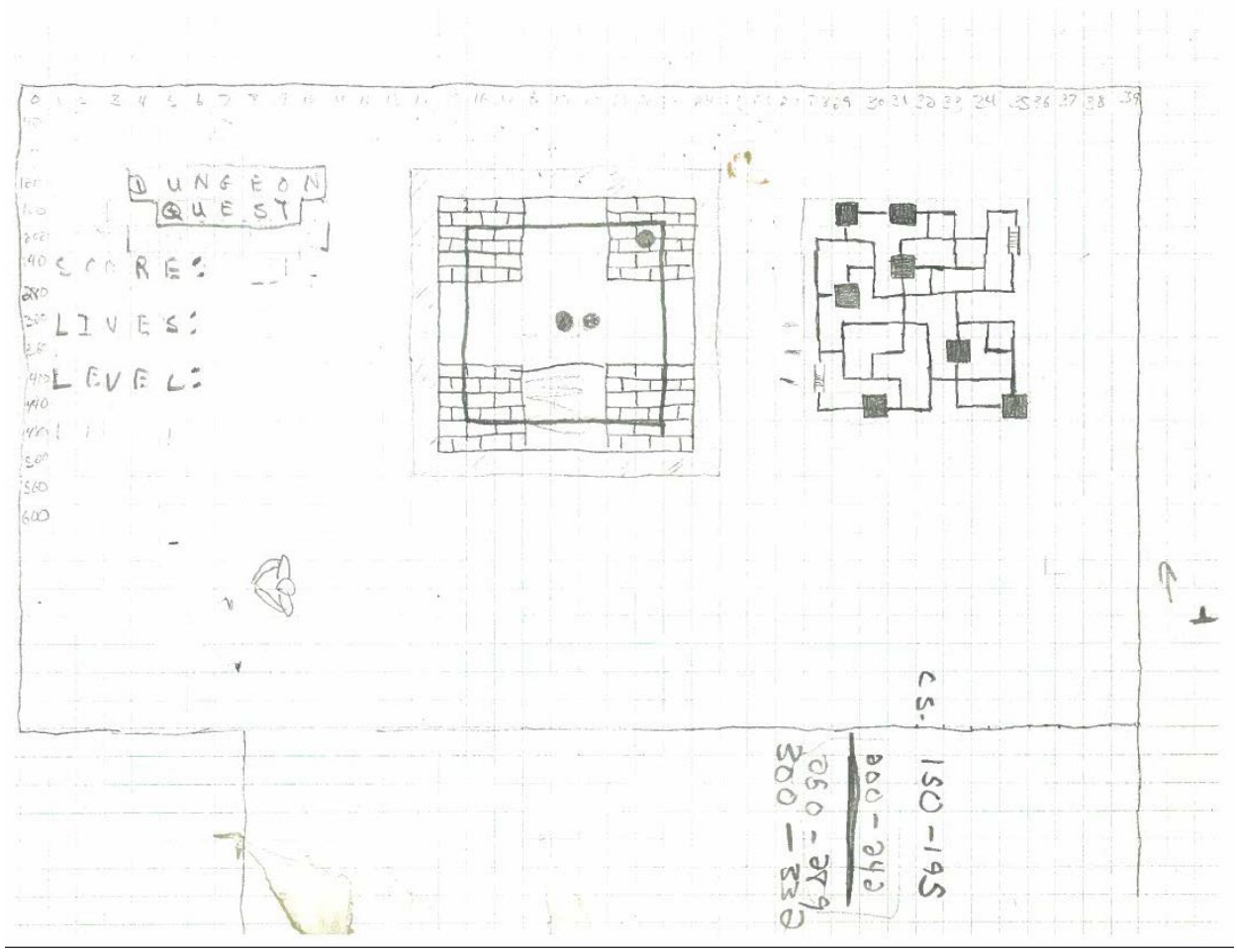
**Recollections**

Kooky's Quest was based on Exidy's Venture arcade game. In Venture, the primary character was called "Winky" and had a smiley face just like the Kooky's Quest main character. Venture also took place in an underground maze, where you wander through halls, enter rooms, and battle the villains in the rooms. All of these game traits are also found in Kooky's Quest.

Kooky's Quest took quite a long time to write and finish. The original name for the game was Dungeon Quest. The game was started before Kooky Klimber, but Kooky Klimber was finished first and set the standard for the Kooky-named series of games which included Kooky's Quest and Kooky Diver. Development got stuck in an almost infinite loop as John Weisgerber tried several techniques to get the

scrolling maze to work smoothly. The wait was worth it, since the scrolling effect looks completely smooth in all four directions, and was implemented through manipulation of text strings in memory.

Kooky's Quest was sold to Antic Magazine for about $200 or $300, and was the first A/W/A program published because Kooky Klimber and Night Rescue were purchased by COMPUTE! Magazine but never published. The A/W/A Team was so excited when the magazine reached the news racks, that they had their parents take them to Fairlane Mall in Dearborn to purchase the magazine the first day it came out.




More information about the technical aspects of this game can be read in the original accompanying article sent to computer magazines, A/W/A Game Write-Ups section of this document.

Antic did a great job with the concept artwork for the game as you can see below!



game of the month

# KOOKY'S QUEST

by ERIC ANSCHUETZ, ROBERT ANSCHUETZ and JOHN WEISGERBER

*A windowed scrolling maze adventure where you fight monsters while searching for the magic key that's your only escape from a multi-level dungeon. Runs in BASIC on any Atari computer with 32K disk or 24K cassette.*

Trapped in an evil dungeon, your only hope of escaping is to descend to the lowest level, retrieve the magic key, and use it to unlock the portal at the uppermost level.

Check the program with TYPO II as you type it in. SAVE the game before you RUN it. Playing Kooky's Quest uses the keyboard and a joystick in port one. There are two skill levels. In the beginner game, you're aided by a displayed map of the dungeon, which is erased for the expert game.

To move from one dungeon level to another, you must obtain enough treasures—rings and chests. You need only one treasure to descend from the first level to the second, but you'll need five to reach each of the deepest dungeons.

Once you have enough treasures to change levels, go to the up or down staircase, shown with a D or U on the map. If you are playing without the map, you can identify a staircase by sound effects, because the stairs aren't in your view.

## VICTORY OVER MONSTERS

Getting treasures is not an easy task. In each room, you will find a treasure and a monster. It is important to get to the treasure before the monster gets it, before you kill the monster or the monster kills you. In each case, you'll lose the treasure.

If you clear all the rooms on any level and don't have enough treasure, you'll be forced to wander that level forever—or until you stop the game.

You have ten lives for the entire game. If you'd like more, rejuvenate yourself at line 610.

You'll be asked to choose a weapon as you enter a room. Available weapons are listed at the bottom of the screen and can be selected with numbers. The axe, mace and sword are swinging weapons—press the fire button and move the joystick from left or right.

The knife, lightning bolt and fireball are throwing weapons. They are used in the same way as swinging weapons—except they continue moving in a straight line until they hit something.

Choose your weapons with care. Each has different properties. You'll find that some require six hits to kill, others only one hit.

Note that while fighting, you can only attack to the left or right, and can't protect yourself from above or below.

You can encounter six types of monsters—vampire bats, giant hobgoblins, wicked skeletons, venomous serpents, decaying mummies and malevolent spiders that inhabit only the lowest level. Your adventurer, Kooky is shown as a happy face symbol.

Instead of escaping the dungeon, you may choose to accumulate points. Treasures are worth five points, defeated monsters ten.

*All three authors are 19-year-old computer science majors at Eastern Michigan University.*

Listing on page 51.

# NIGHT RESCUE (1984)

http://www.atarimania.com/game-atari-400-800-xl-xe-night-rescue_31439.html




**Game Play**

Night Rescue puts you on a daring attempt to rescue civilians with your hot air balloon before they are taken prisoner by the invading forces. The player must navigate through a lengthy journey through a

city without crashing into any of the buildings or ground, avoid obstacles, and rescue as many humans as possible. The enemy consists of tanks, rockets, helicopters, and a manic warplane. You must maneuver through a city-scape and reach your ultimate destination where you rescue the last hostages. Each rescued hostage gives you 100 points.

The following description of the gameplay was written by the A/W/A Team to be submitted to a computer magazine along with the game.

> *Night Rescue is a game in which you guide your balloon over enemy territory on a secret mission to rescue your country's spies who have infiltrated the enemy's city. The journey, which begins over the mountains on the outskirts of the city, is extremely dangerous because any contact with enemy terrain or the enemy reconnaissance plane and helicopters results in the loss of one of only five balloons. In addition to this, the enemy has planted double-agents who will destroy your balloon if you attempt to rescue them. Upon reaching the outer edge of the city you must make a soft landing on the top of the headquarters of the resistance, at which point you will begin another run above the city.*
>
> *The balloon is controlled by moving the joystick left, right, up, and down to move in the corresponding direction. The balloon's position may be "fine-tuned" by pressing the fire button while moving the joystick left and right. Moving the balloon up and down is made treacherous by the forces of inertia acting upon it. To land on the resistance's headquarters you must be both scrolled and "fine-tuned" all the way to the right. Scoring in Night Rescue consists of 100 points for each rescued agent plus a bonus for reaching the resistance's headquarters which increases for each new set. At the end of the game you may play again by simply pressing the START key.*

**Recollections**

Night Rescue was actually based on four different games. The first two games were home computer games called Protector, by Synapse, and The Tail of Beta Lyrae, by Datamost. Both of these games were clones themselves of the third game it was patterned after, Williams Defender. Protector, The Tail of Beta Lyrae, and Defender were side-scrolling shoot-em-ups. Night Rescue copied the city-scape environment of the home video games, while also incorporating the rescue objective of Defender where the main player had to land on top of the characters to be rescued at the bottom of the screen. The fourth game that Night Rescue was patterned after was Choplifter, which was initially developed for the home computer scene and, in a very rare turn of events, was subsequently released as an arcade game. Choplifter was also a side-scroller, featured a helicopter as a main player, and also had a mission where characters at the bottom of the screen had to be rescued. Night Rescue, however, was not a shoot-em-up, but rather a game of skill and timing.

Night Rescue had a beautifully rendered city-scape of futuristic buildings and made use of the many-colored palette of the Atari 8-bit computers. The game was rendered in an Atari Graphics mode that

wasn't available directly from BASIC, but could be accessed by writing various memory locations within the Display List with the correct values to enter the "hidden" mode. This mode added additional colors to each character. There was even a special trick that "borrowed" a color from Player/Missile graphics to be used as an additional color in each of the characters.

Mapping out the city-scape on graph paper was a creative process. Instead of black and white (1's or 0's) colors for the bit-mapped 8x8 characters, each character in this special graphics mode could have 4 or 5 colors. The city-scape was several screens wide, and was rough-scrolled through 8-bits at a time. Unlike many side-scrollers that force the player in one direction, usually left to right, the Night Rescue player could control the direction of horizontal scrolling with the joystick. The player's mission was to go from the left side of the city-scape to the right-side, through several screens of buildings and caves, but at any time the player could turn around and go back left to rescue more humans for points or avoid obstacles like the flying plane.

Originally, since the game was set in Germany during WWII, a Nazi flag was used as a recurring character in the landscape. Eric, Robert, and John realized that a Nazi flag may not be politically correct if the game were to be published, so they changed the flag to be a modern Germany flag with three colors. They also added extra redefined characters to change the helicopters' orientation from left to right.

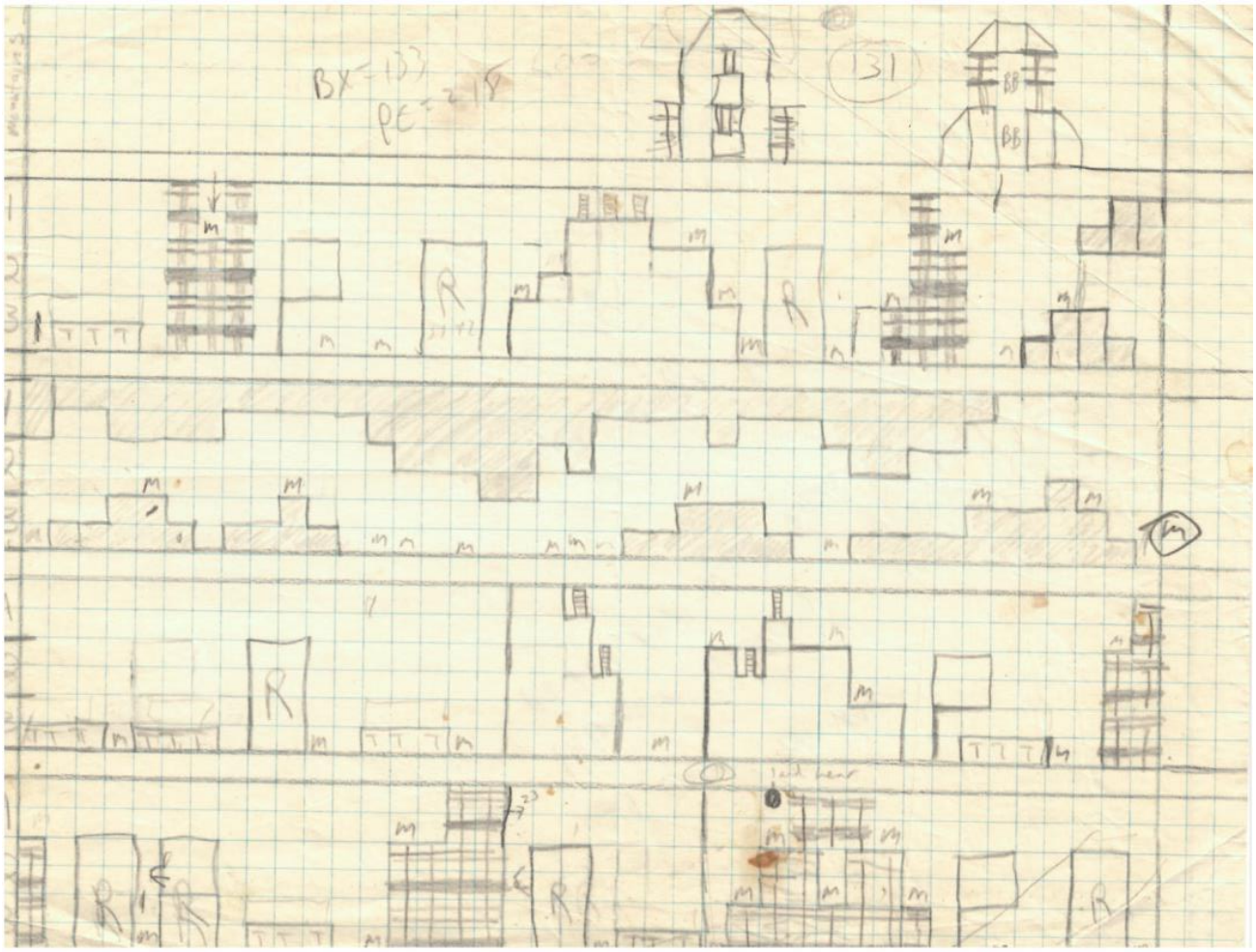
The main player controlled a hot air balloon. The balloon was moved left/right/up/down with the joystick. When the balloon moved left and right, the terrain scrolls in the opposite direction. A really unique method of control was added to fine-tune movement. This was accomplished by pushing the fire button while moving left and right. If the fire button was pushed while moving, the balloon moved 1 pixel at a time left or right, and the background didn't scroll. This maneuver was essential to master in order to rescue some of the humans tucked away in very difficult to reach areas in the gamescape. The balloon movement was also affected by very realistic vertical momentum. The longer you held the joystick in a vertical direction, the more the momentum was built up and it takes a while to counteract this momentum to slow down and go in the opposite direction. This momentum, and the need to sometimes fine-tune movement with the fire button, made maneuvering in tight quarters very difficult.

While the balloon mostly travelled from left to right, an out-of-control plane obstacle constantly got in the way of the player as the plane travelled from right-to-left. The plane actually moves in a sine-wave pattern, and the A/W/A Team initially actually used a mathematical equation to control the plane. The equation's sinusoidal movement was taxing on BASIC and slowed it down, so another algorithm was used that moved the plane up or down a pixel as it travelled until it hit an upper or lower limit and reversed direction. The plane stuck to the top-third of the screen as it moved from right-to-left. When the plane departed the left side of the screen, another one soon followed from the right. The plane was always in the way of the player, and constantly forced the player to adjust positions.

One interesting part of the game that gave the player an option was the cave section. The player had the option to navigate through a tight cave which was loaded with lots of humans to rescue for points, or the player could take the somewhat easier route above the caves but always in the path of the plane. This decision was really equally weighted one way or another, and the decision really depended on a

risk/reward strategy and the skill of the player. The gameplay of Night Rescue was very good – it was easy to get the hang of and very hard to master. Once the player reached the base at the right-end of the city-scape, he was rewarded with bonus points and started all over. The next time through, the plane flew a little faster and it was a little harder to complete the mission.

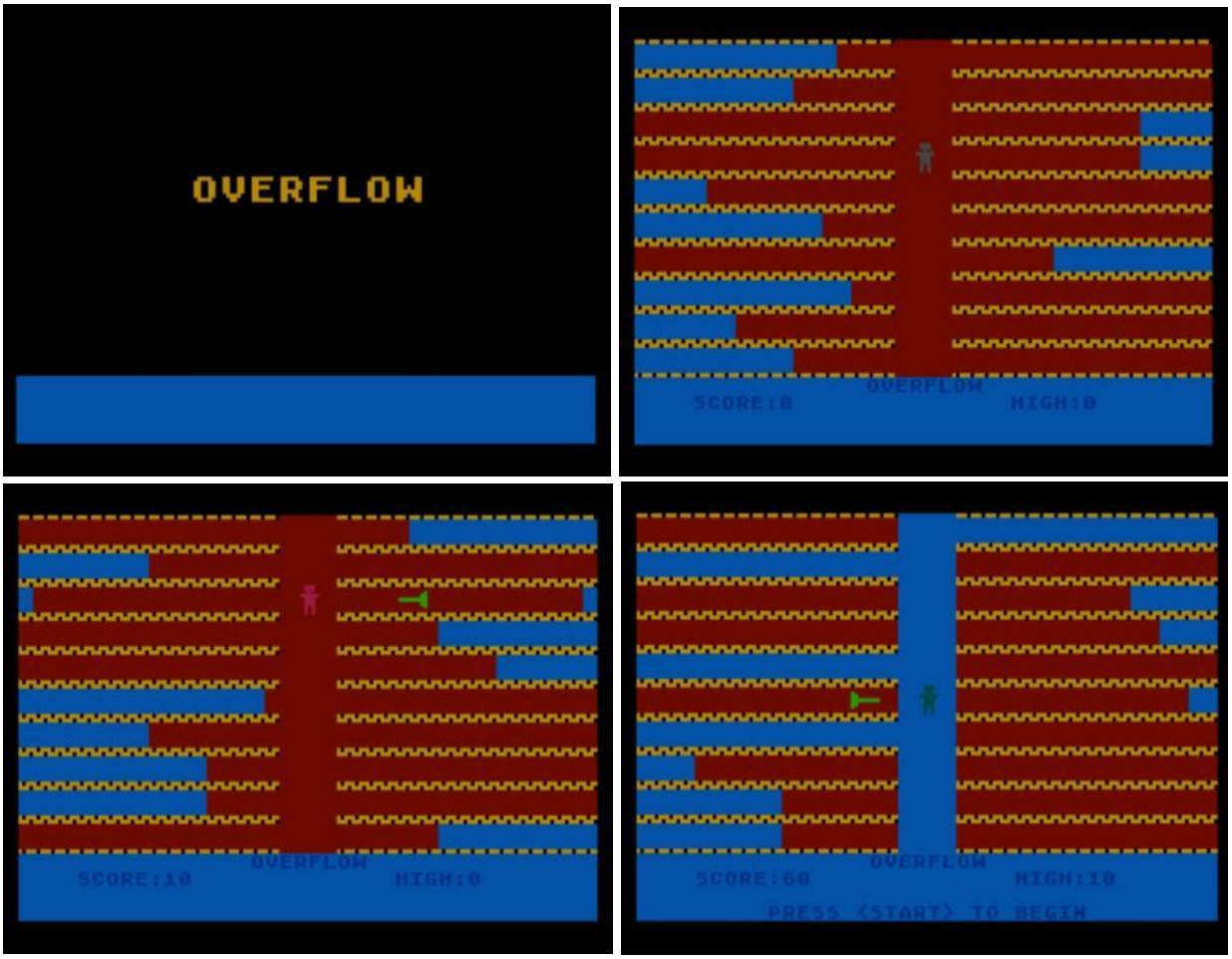
Night Rescue was the second game that the A/W/A Team sold to COMPUTE! Magazine, again for $400. Like Kooky Klimber before it, for some reason the game was never published. One memory that John has to this day is that he always felt that he was (partly) to blame for Kooky Klimber and Night Rescue not getting published. The A/W/A Team believed Night Rescue was more polished and valuable than Kooky's Quest and wanted more money than the $400 offered for Kooky's Quest. John had a phone conversation with a lady at COMPUTE! Magazine that turned kind of hostile. He finally agreed to the $400 anyway when they lady said "take it or leave it". John was hardly diplomatic at that age! They eventually published "Crazy Climber" games for other platforms, but never published either Kooky Klimber or Night Rescue, so John always wondered if that was a snub due to that phone call!



More information about the technical aspects of this game can be read in the original accompanying article sent to computer magazines, A/W/A Game Write-Ups section of this document.

# OVERFLOW (1985)

http://www.atarimania.com/game-atari-400-800-xl-xe-overflow_3847.html




**Game Play**

The following description of the gameplay was written by the A/W/A Team to be submitted to a computer magazine along with the game.

> *In Overflow, you're a plumber trapped in the central conduit of a network of pipes. Water is advancing relentlessly toward you through the secondary pipes. If the water gets far enough, it will flood the main pipe, drowning you and ending your game. This is a situation you (as a devout plumber) obviously want to avoid. Luckily, you can temporarily reverse the flow of water in a pipe by hurling your plunger down the pipe at the onrushing column of water. Scoring is based on the fact that when the plunger makes contact, in addition to making the water recede, you earn ten points. Controlling the flow of water is simply a matter of moving up and down in the main pipe and throwing your plunger*

*down the pipes closest to the point of overflowing. Unfortunately, even if you manage to gain control of the situation, at five-hundred points the width of the main pipe doubles, making your job all the more difficult. Movement is controlled from a stick in port one. To move up and down in the central pipe, push the stick forward and back. To throw your plunger, push the stick left and right.*

The following gameplay instructions for Overflow were published in Antic Magazine.

*Dear Miss Goodmanners:*

*Is there a correct way to stem a rising tide of bathroom back-ups while entertaining distinguished guests in my 49-room mansion?*

*Concerned*

*Dear Concerned:*

*Miss Goodmanners prefers the term water closet... Sadly, overflow problems are part of our modern age and must be dealt with firmly and quickly. Miss Goodmanners would never commit the faux pas of embarrassing guests during a sumptuous banquet by confronting them with a mass of raw sewage.*

*Miss Goodmanners would award you credit for decisively hurling your plumber's helper at each of the overflowing water closets, while tactfully running between pipelines without scaring the guests. Of course, sewage problems tend to get worse as a party wears on-especially if guests discover that the duck pate has disagreed with them.*

*With apologies to our favorite etiquette columnist, this program uses players for the plumber, plunger and central pipe. All vertical motion is produced using a string printed at the correct place in memory. The main graphics mode is ANTIC 5.*

*An unusual technique here is the use of basically the same memory address for every line of the playfield graphics. Each line has its own LMS and the high byte on every one is set to the same page. The water is moved back and forth by incrementing the low byte on each LMS. The increment for each line is held in a one-dimensional array. If a column of sewage is hit by the plunger, the increment's sign is simply reversed.*

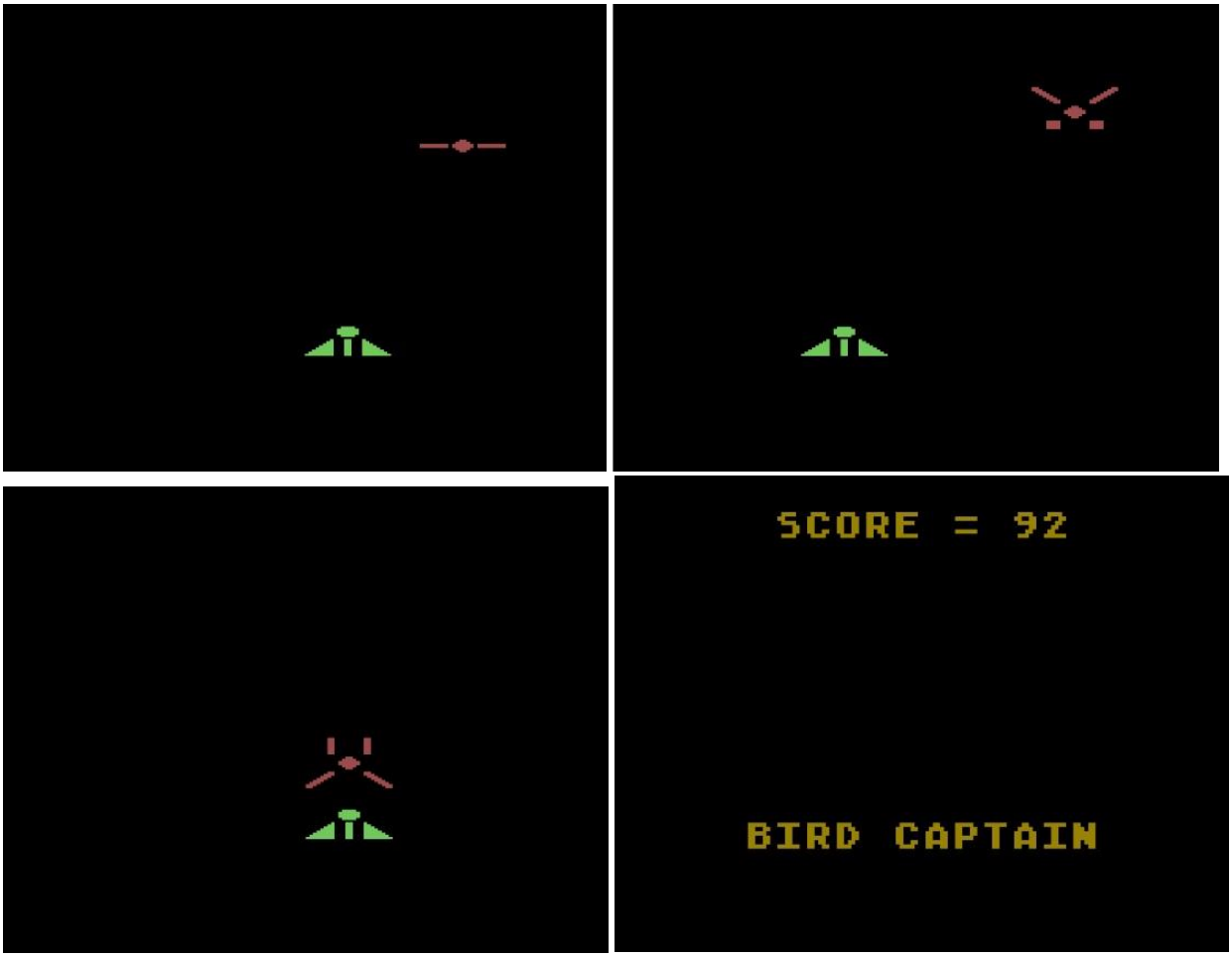
**Recollections**

This game was written entirely by John Weisgerber, but he graciously consented to credit the game to A/W/A authorship and even split the $200 to $300 from Antic Magazine for its publishing rights. This program uses Player/Missile graphics for the plumber, plunger and central pipe. All horizontal motion of the water flowing through the pipes is produced using a string printed at the correct place in memory. These were some pretty sophisticated techniques in the early days of BASIC programming.

More information about the technical aspects of this game can be read in the original accompanying article sent to computer magazines, A/W/A Game Write-Ups section of this document.

# PHOENIX (1982)

http://www.atarimania.com/game-atari-400-800-xl-xe-phoenix_31474.html




**Game Play**

Phoenix had the player control a black and white ship at the bottom of the screen made up of multiple text characters, and a single large bird with flapping wings that dodged all over the screen. A press of the fire button on the joystick produced a vertical shot aimed at the bird. If the bird was hit, the player scored a point, and the bird would turn upside down and fall straight to the bottom and the player would try for the next bird. After 9 birds were hit, the game ended and the player was rated as "Phoenix Commander", "Bird Captain", "Butterfly Netter", or "Bird Watcher" depending on the score.

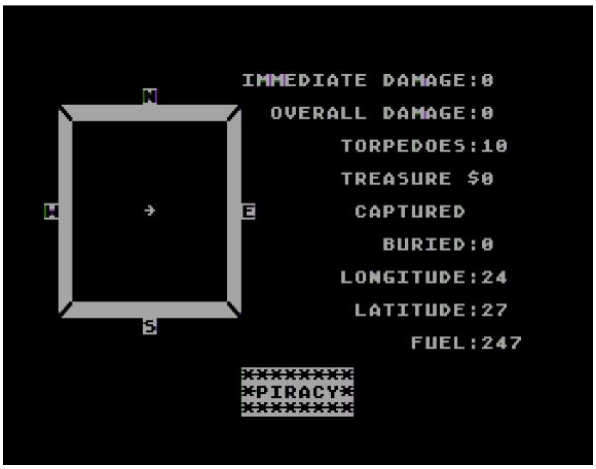
**Recollections**

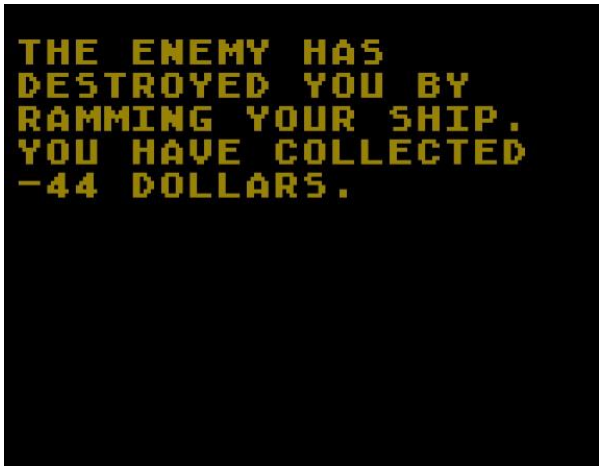
Phoenix was the very first video game written by Robert Anschuetz of the A/W/A Team shortly after they got their Atari 400 for Christmas in 1981. The game was implemented in a simple text mode using an alternate built-in Atari font library called ATASCII "control" characters. The game was based on an example program called "Seagull Over Ocean" from the Atari BASIC Reference Manual that

demonstrated a seagull flapping its wings and flying around the screen. There were no special graphics techniques used to develop the game. This game had been missing for almost 30 years, because the A/W/A Team did not back the game up to a PC using an SIO2PC cable in the early 1990's. The game was retrieved from an original 5 ¼" floppy disk in 2017 and finally saw the light of day again.

# PIRACY (1982)

http://www.atarimania.com/game-atari-400-800-xl-xe-piracy_31476.html




**Game Play**

In the game of Piracy, the player navigated a ship using a map. The goal was to collect treasure before your immediate or overall damage was too great, you were rammed by a ship, or you used too much fuel.

**Recollections**

Piracy was a simple game that was written in Atari Graphics 0 text mode. There were no special graphics techniques used to develop the game. This game had been missing for almost 30 years, because the A/W/A Team did not back the game up to a PC using an SIO2PC cable in the early 1990's. The game was retrieved from an original 5 ¼" floppy disk in 2017 and finally saw the light of day again. This game was an inspiration for the much superior Piracy II that was written by the A/W/A Team several years later.

# PIRACY II (1986)

http://www.atarimania.com/game-atari-400-800-xl-xe-piracy-ii_31441.html






**Game Play**

Piracy II is a multi-screen epic battle over the high seas. The object of the game is to confront a pirate ship at sea by sailing through the ocean and finding ships that are protecting treasure located on any of many nearby islands. Once a pirate ship has been encountered, a mini-game is launched where you control the pirate ship at the bottom of the screen and you must shoot a pirate guarding his ship. The ship at the player's control must dodge a constant bombardment of cannon balls aimed your way.

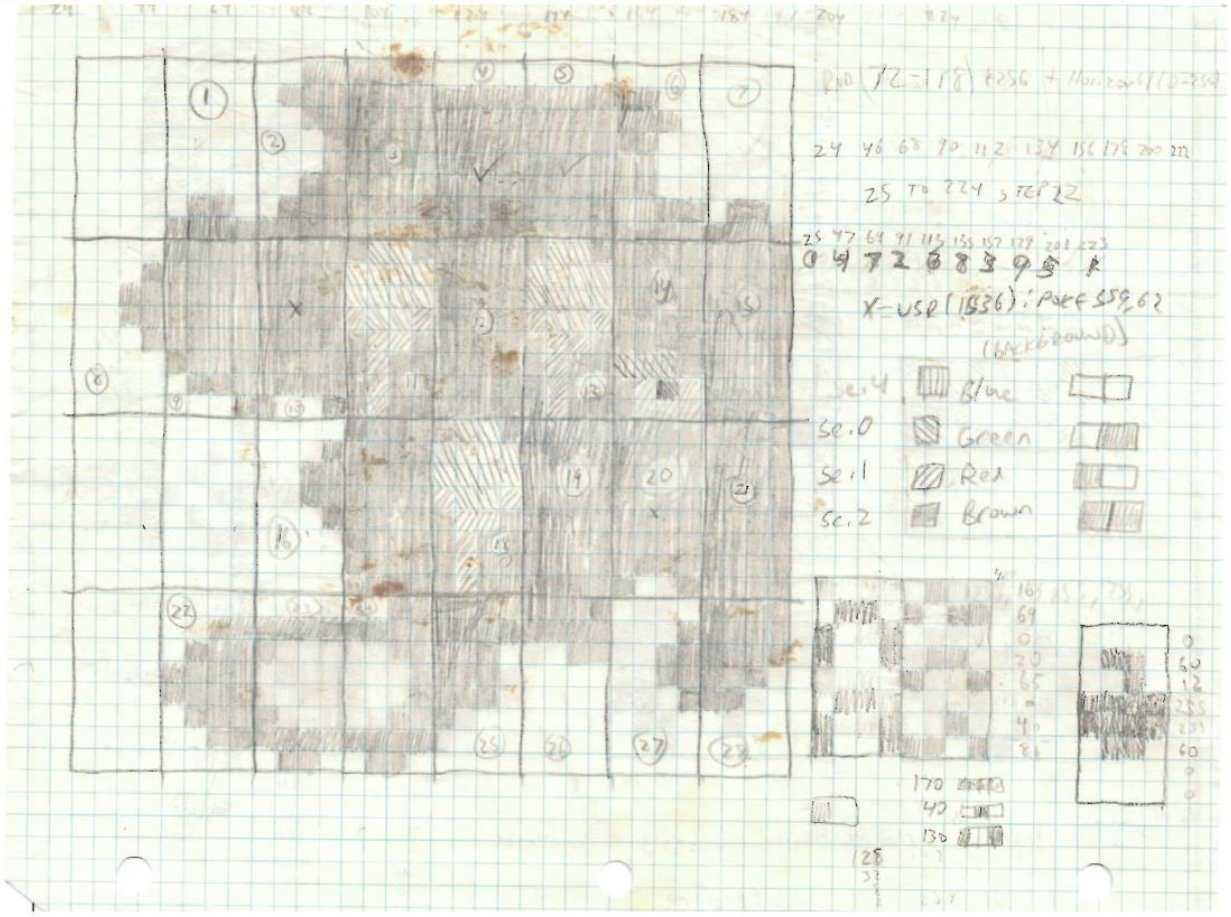
After the pirate on the ship has been defeated in the mini-game, the player is returned to the open seas. At this point, the player may land the boat on the island the ship had been protecting, and another mini-game is launched. This mini-game forces the player to navigate through a moving maze of pirates as he makes his way to the treasure chest located on the far side of the island. The key to this mini-game is observing the pattern of movement by the pirates on the island. If the treasure chest is successfully reached, the player is returned once again to the open seas to seek out additional treasure by confronting other ships guarding other islands.

**Recollections**

Piracy II is a sequel to the original Piracy game that was written several years before Piracy II. Piracy II was written by Eric and Robert Anschuetz. The game of Piracy II is a multi-stage game, which is very rare for BASIC games in the 1980's. The game uses fine vertical and horizontal scrolling, which is a nearly impossible using Atari BASIC. The fine scrolling was made possible with Assembly Language routines written by Eric Anschuetz, and called using the BASIC USR command. Even so, there was occasionally a flicker when the scrolling became too much for the processor. Eric and Robert did not have an Assembler cartridge, so all of the assembly language was "hand assembled" by Eric. This is the only primarily BASIC game (with Assembly added for scrolling) that they know of that has a smooth scrolling horizontal and vertical multi-screen display.

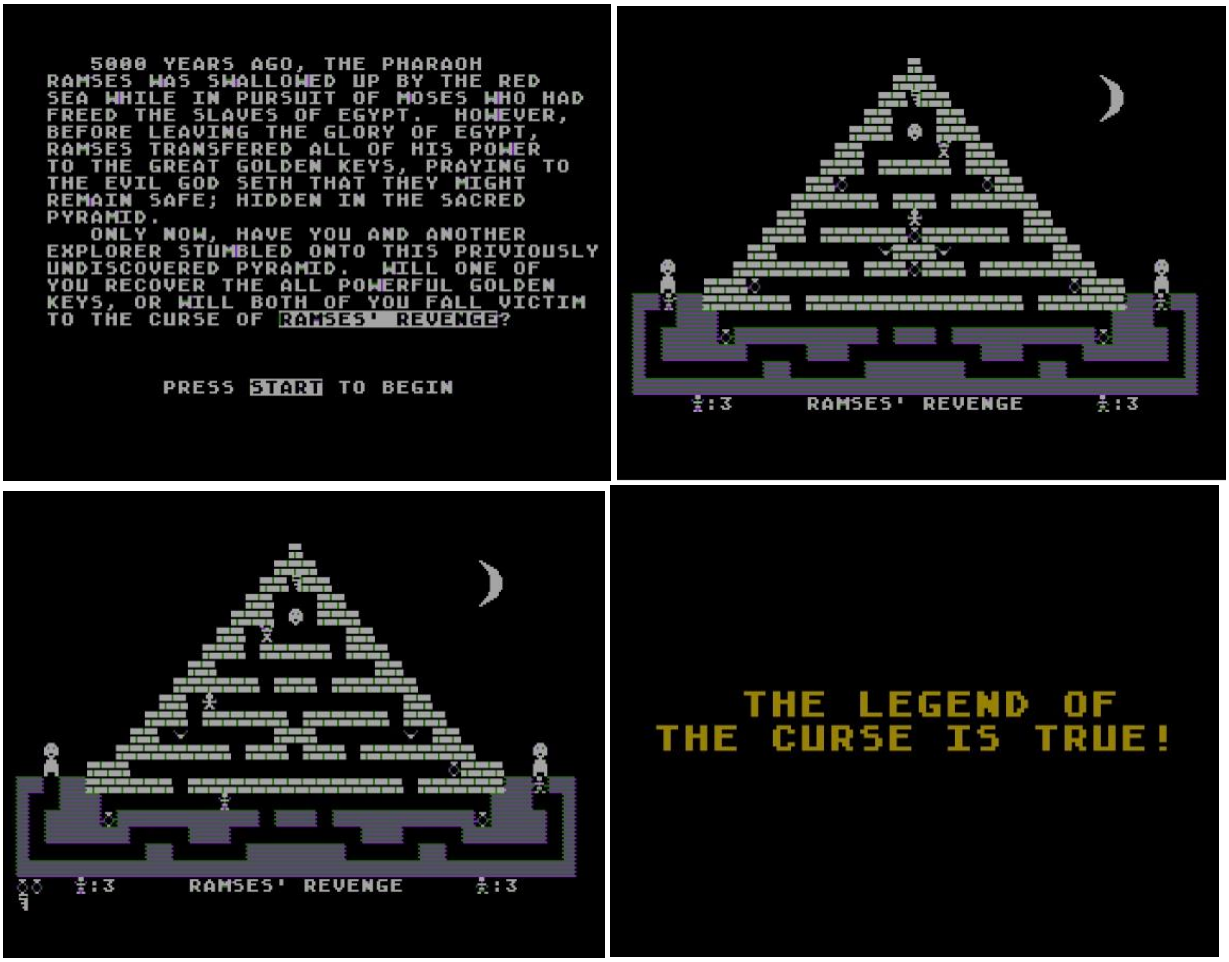
The ship is a Player/Missile graphic, but most of the other graphics are based on redefined characters. The pirate ship mini-game scene is very reminiscent of the ship-battle scene in the Pirates of the Caribbean ride at Disney World, and was probably influenced by it. In fact, the whole entire game was

probably influenced by that ride since the A/W/A Team had just gone to Disney World a couple years before the game was written.



# RAMSES' REVENGE (1983)

http://www.atarimania.com/game-atari-400-800-xl-xe-ramses--revenge_31475.html




**Game Play**

5000 Years ago, the Pharaoh Ramses was swallowed up by the Red Sea while in pursuit of Moses who had freed the slaves of Egypt. However, before leaving the glory of Egypt, Ramses transferred all of his power to the great golden keys, praying to the evil god Seth that they might remain safe; hidden in the sacred pyramid. Only now, have you and another explorer stumbled onto this previously undiscovered pyramid. Will one of you recover the all powerful golden keys, or will both of you fall victim to the curse of Ramses' Revenge?

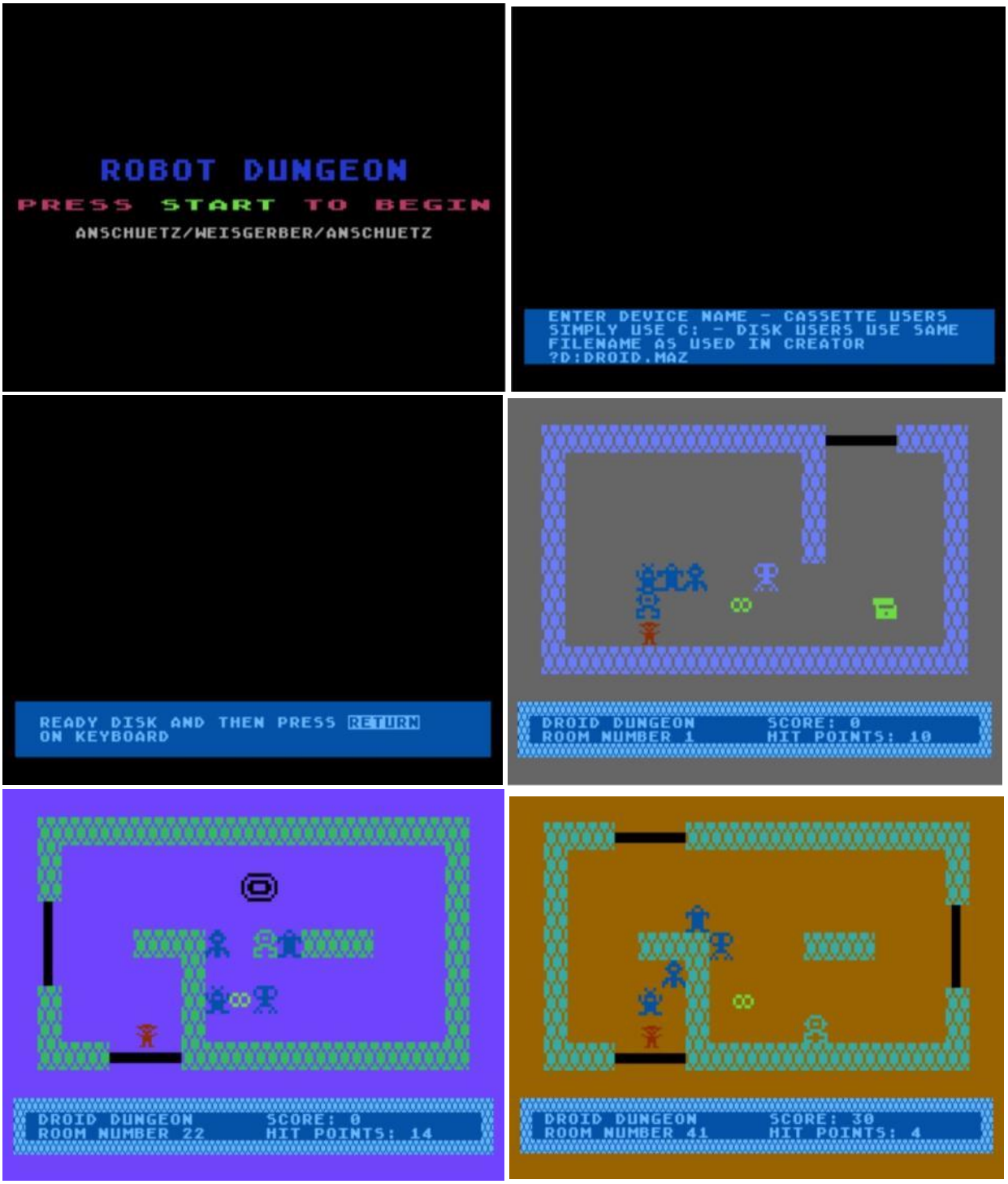
Two players start at opposite ends of a pyramid, and race their way underground and then to the top chamber. Along the way, moving cobras, bats, and other obstacles get in the way. The first player to the top wins and is greeted by a musical melody.

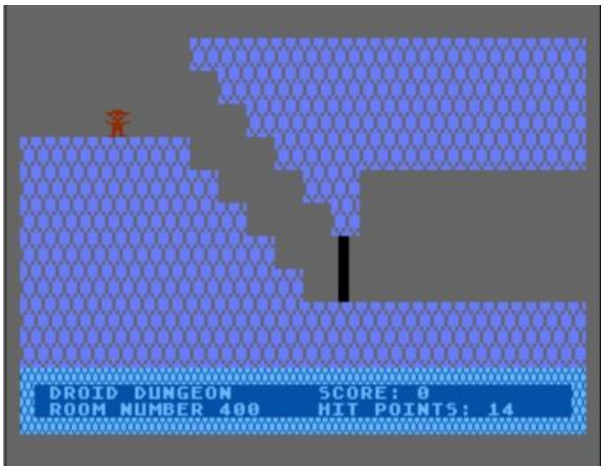
**Recollections**

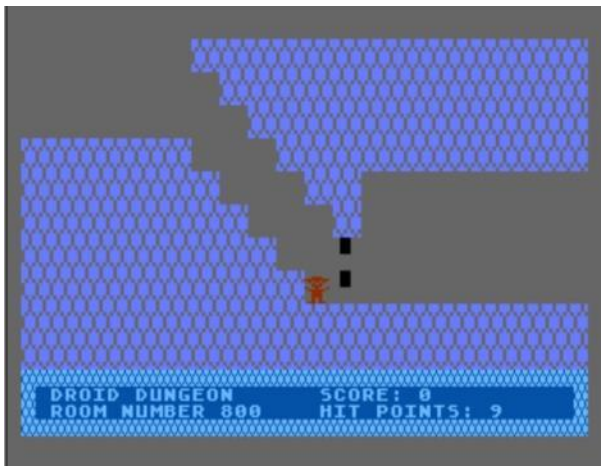
Ramses' Revenge was the first quality playable game that the A/W/A Team wrote. This game was loosely based on Konami's arcade game Tutankham as well as Stern's arcade game Lost Tomb. The A/W/A pyramid-based game was written in the Atari's text mode called "Graphics 0" in BASIC. It made extensive use of re-defined characters that allowed a new bit-mapped character set to replace the built-in characters of the Atari computers. This game was lost for about 30 years because it wasn't transferred to a PC with the SIO2PC device in the 1990's. In fact, the A/W/A Team didn't even remember the name of the game and referred to it as "the pyramid game." In 2017, the A/W/A Team found the game on the backside of a floppy disk and recovered it again with SIO2PC.

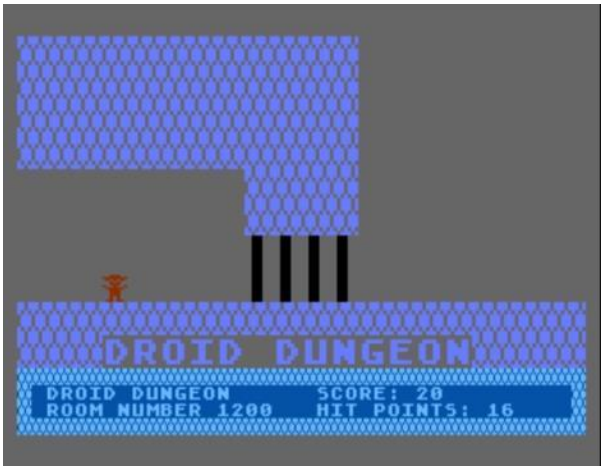
# ROBOT DUNGEON (1985)

http://www.atarimania.com/game-atari-400-800-xl-xe-robot-dungeon_4411.html















**Game Play**

The following description of the gameplay was written by the A/W/A Team to be submitted to a computer magazine along with the game.

*ROBOT DUNGEON is known for its deadly robots, electrified walls, and teleports. Rumor has it that no one has ever escaped alive! Nobody knows for sure, but the best estimations are that ROBOT DUNGEON is comprised of 3 levels of 400 rooms each, arranged in a 20x20 format.*

*The object of the game is to traverse from the lower-left room of each level to the upper-right room, all while amassing as many lives and as high a score as possible. Upon reaching the upper-right room of the 400 rooms on Level 1, you must touch the staircase and you will then procede automatically to walk down to the next most treacherous level, Level 2. At this point there is an animation sequence that will give you time to relax. Sit back and enjoy your figure as he jumps down the stairs, approaches the door, and slips through just in time!*

*Similar to what you did on Level 1, you must also reach the upper-right room of Level 2 (labeled room #800). Touch the staircase here and you will be rewarded with viewing the animation sequence once again, and will proceed to the very dangerous Level 3!*

*If you manage to guide yourself to the upper-right room of Level 3 (labeled room #1200), and touch the escape key, you will be greeted with an ending animation sequence showing yourself leaving the dreaded ROBOT DUNGEON (and as you will see, apparently just in time!).*

*ROBOT DUNGEON is no ordinary dungeon, either! No, ROBOT DUNGEON is rather a 3 level MAZE with only 1 escape route possible for each level. Therefore, it is heartily recommended that you map each level.*

*Throughout ROBOT DUNGEON there are various treasures and traps which with you must be acquainted with. The most abundant of these figures is the "SYMBOL OF LIFE". (It looks like a sideways figure eight). Touching this symbol will up your hit point total by one. The hit point total (which is found on the lower-left of the score window) decreases by one every time you touch a robot, touch a wall, or touch a teleport (more on teleports later). It is very important to get the "SYMBOL OF LIFE" in each room or you will have a very short adventure in ROBOT DUNGEON, indeed!*

*A symbol closely related to the "SYMBOL OF LIFE" is the "TREASURE CHEST". Within the "TREASURE CHEST" lies 5 hit points. Needless to say, don't pass up many of these!*

*Another symbol is the "ASSORTED GOLD". This pile is worth 250 points! Akin to the "ASSORTED GOLD" is the "DIAMOND", which is worth 1000 points!*

*Perhaps the most useful symbol is the deadly "ROBOT ZAPPER". This symbol looks like a zig-zag line and kills every robot in a room when touched!*

*Something to definitely stay away from are the dreaded "TELEPORTS". These are shaped like black bits and bring you back to the first room of the level which you are on! They also cost you one hit point! Stay away from these at all costs!*

*You are not defenseless as you wander ROBOT DUNGEON. You are equipped with a pistol which can be aimed in any direction. To shoot, point your joystick in a direction and press the fire button. The killing of each blue robot is worth 10 points. Notice how I said blue robot. The orange ones are indestructible! On Level 1, there is one orange robot and 4 blue robots in each room. On the second level, there are2 oranges and 3 blues. And on the third level, there are 3 oranges and only 2 blue ones!*

*You will note that as you pick up symbols, they are gone forever but robots are resurrected each time you enter or reenter a room.*

*By pressing any key during the course of play, you can freeze the action. Press another key and play resumes. You may also save the game in progress by pressing any one of the console keys (START, SELECT, or OPTION). Note that on a disk system, saving a game will overwrite the previous version of your maze. If you want to have several different stages of your progress, you will have to change disks. Be sure to save your game very frequently!*

*When your hit point total reaches zero, your game is over. Pressing the start button will give you the choice of loading in a new dungeon or replaying the old one at the same point it was at when you began. If it was a saved game in progress, you start where the saved game started. However, all symbols that you had gotten since you began the session will now be gone.*

The following Robot Dungeon gameplay instructions were published in Antic Magazine in a special Disk add-on. This was not the exact write-up that the A/W/A submitted with the game, but most of these instructions were borrowed from their original write-up.

> *ROBOT DUNGEON is known for its deadly robots, electric walls, and teleports. Rumor has it that no one has ever escaped! Nobody knows for sure, but most believe that ROBOT DUNGEON is comprised of three levels of 400 rooms each. Rooms are arranged in a 20 x 20 grid on each level. The object of the game is to traverse from the lower-left room of each level to the upper-right room while collecting as many lives and points as possible. Upon reaching the upper-right room of Level 1, touch the staircase to get to Level 2. Here, you'll see a cartoon that will give*

*you time to relax. Sit back and enjoy the figure as he jumps down the stairs and slips through the door just in time!*

*Complete Level 2 the same way you completed Level 1. (The upper-right room is #800). Remember to touch the staircase to see the cartoon and advance to Level 3. If you manage to get to the upper-right room of Level 3 (room #1200), touch the key and you'll see another cartoon! ROBOT DUNGEON is no ordinary dungeon, either! It is a 3-level maze with only one escape route on each level. (We recommend you map each level.) ROBOT DUNGEON has many treasures and traps which with you must be acquainted. The most abundant of these is the "SYMBOL OF LIFE." (It looks like a sideways figure eight). Touching this symbol will increase your hit point total by one. The hit point total (on the lower-right of the score window) decreases by one every time you touch a robot, touch a wall, or touch teleport (more on teleports later). It's very important to get "SYMBOL OF LIFE" in each room or you'll have a very short adventure! A symbol closely related to the "SYMBOL OF LIFE" is the "TREASURE CHEST". Each one holds 5 hit points. Another symbol is the "ASSORTED GOLD". This pile is worth 250 points! The "DIAMOND," like "ASSORTED GOLD" is worth 1000 points. The most useful symbol is the "ROBOT ZAPPER." It looks like a zig-zag line and kills every robot in a room when touched! Stay away from "TELEPORTS." These are shaped like black pits and return you to the first room of the current level. They also cost you one hit point.*

*You are not defenseless as you wander ROBOT DUNGEON. You have a pistol! Aim and fire it with your joystick. Each blue robot you destroy is worth 10 points. Orange robots are indestructible. Level 1 has one orange robot and four blue robots in each room. Higher levels have more orange robots and fewer blue ones! NOTE: When you pick up symbol, it's gone forever, but robots are resurrected each time you enter a room. By pressing any key during the course of play, you can freeze the action. Press another key and play resumes. To save a game in progress, press a console key (START, SELECT, or OPTION)!*

*Saving a game will overwrite the previous version of your maze. If you want to save several different stages of your progress, you will have to change disks. Be sure to save your game very frequently! When your hit point total reaches zero, the game is over. Pressing the START button gives you the choice of loading a new dungeon or replaying the old one from the last SAVE. NOTE: All symbols of the previous game are lost.*

*To play ROBOT DUNGEON, you first need to make a maze with the MAZE CREATOR PROGRAM, DUNGEON.CRT! This program will ask you to name your dungeon. This is NOT the disk filename. Type in a name for the dungeon, and press return. Next, you are asked to give a disk filename for your dungeon data. After pressing [RETURN], there will be a 15-20 minute wait for the maze to be created. Don't worry about the numbers being printed. For those of you who are interested, the first column stands for the level, the second is the room number, and the third and fourth are used for determining exits. Once the program is through, you can RUN the main game, ROBOTS.BAS. When the computer asks you for a filename, type in the same name you gave it in the Maze creator program. Load the maze by pressing RETURN several times. You do not need to make a new dungeon every time you play, use the same one over and over until you complete it. When you're done with one maze, you can create a new one!*

## Recollections

Robot Dungeon features a hexagonal maze generation game that borrows heavily from game elements of the arcade game Berzerk, as well as Adventure for the Atari 2600. John Weisgerber discovered a BASIC maze-generating algorithm he read in the Ypsilanti High School library in the June 1981 issue of Creative Computing in an article called “Bee Amzazed.” The “Bee Amazed” algorithm generated a maze with hexagonal rooms that when pieced together looked like a honeycomb, with the entry point in the lower-left corner of the maze, and a single exit point at the upper-right corner of the maze. Each room in the generated maze has 1 to 6 exits leading to adjacent rooms. John used the basis of this algorithm to create a maze generator at the heart of the game. Instead of representing each room as a six-sided hexagon, each room is represented as a rectangular shape with the two potential doors on both the top of each room, two potential doors on the bottom of each room, and one potential door on each on the left and right sides of each room. Therefore, a total of 6 potential doors in each room corresponded to the 6 sides of a hexagon that was part of the maze. The rooms fit together like staggered puzzle pieces. When you exit a door in the upper right of one room, you enter the door in the lower left of the room above.

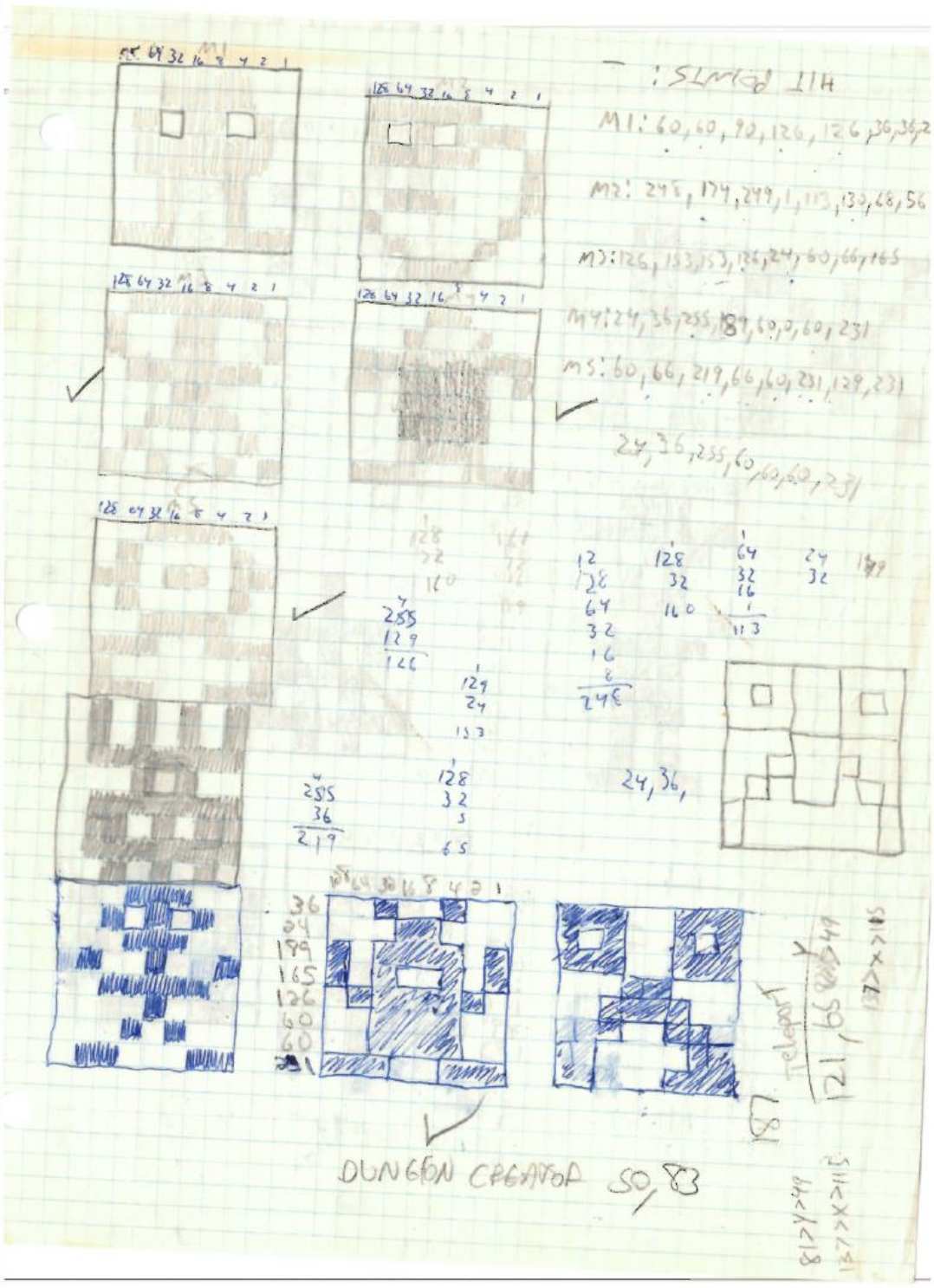
Robot Dungeon was a really long game for the player to complete. There were three levels of 400 rooms, for a total of 1200 rooms. To reward the player, the A/W/A Team actually programmed cut-scenes at the end of each level with humorous cartoons much like the Pac-Man intermissions. Besides the cut scenes, the programming of this game provided two additional advanced features for a BASIC game. The game offered a “pause” feature and it also provided an “auto save” feature.

The robot movement algorithm was written so that with each iteration of the main processing loop, the robots compared their location to the location of the player and headed toward him. The robots were large redefined characters. The main player was a Player/Missile graphic, as was the shot that the main player could shoot at the monsters.

A separate maze generation program was written to accompany the main program. This needed to be run to create a maze, prior to being able to start the game. The maze could be saved to either cassette or disk. When the A/W/A Team wrote Robot Dungeon, they hadn't yet bought a disk drive. Knowing that there was a limitation of filename length in Atari DOS 2.0, but not knowing if it was 8 characters or 6 characters, the A/W/A Team instructed the player to name the file as "DUNJIN.DAT" instead of "DUNGEON.DAT" just to be safe.

In order to remember the contents of each of 1200 rooms, a novel technique was developed by John. By using just one 8-bit byte for each room, individual bits represented which of the treasures or obstacles were in the room. This saved up to eight times the memory a traditional technique would have used. This game was also pretty buggy. Because the characters were jumping around, the algorithm for computing whether a missile hit a monster was not always accurate and the missile would just pass through the monster. Another bug would occasionally allow the player to go right through a wall and be trapped outside the playfield. This was because the comparison of a player collision with a wall or door was too slow to run every loop. One day, Eric and Robert received a phone call from Antic magazine where a game evaluator was stuck outside of a wall and asked what to do! The good news with that phone call was that it was a positive indication that the game was being considered for publication. Ultimately, it was published in a "Add-On Disk" edition that provided a floppy disk with the magazine to avoid typing in the complete program.

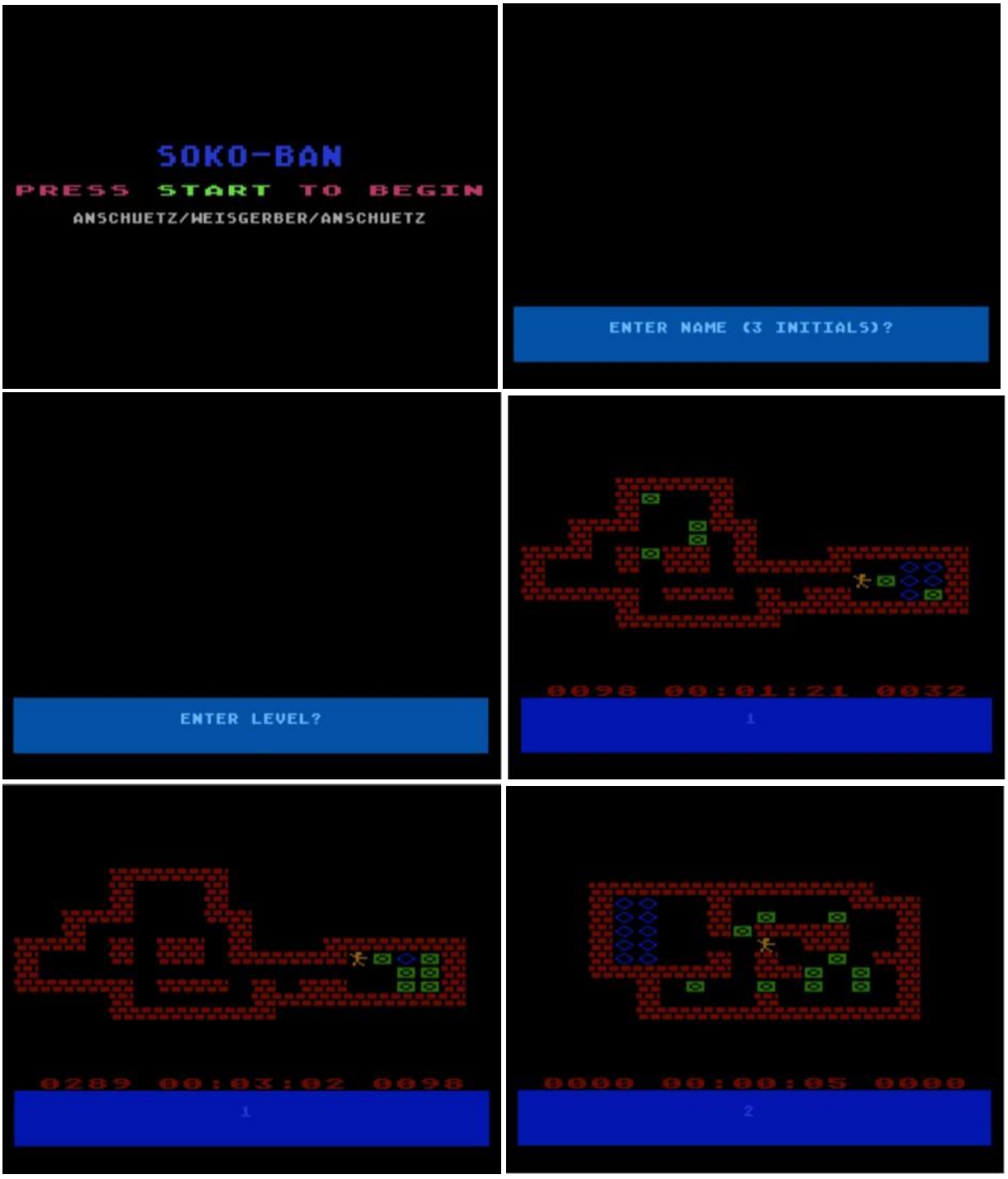
One of the issues that prevented good gameplay with this game was the massive amount of rooms that must be traversed to complete the game. In the end, the A/W/A Team should not have made each maze 20x20. That is just too large to allow for satisfying game play. In 2017, Eric addressed this issue and updated both the main game and the maze creator to allow the user to create any maze size from 5x5 up to 20x20, instead of the previous 20x20 maze. Eric also modified the game to allow a cheat to walk through walls and monsters so that you can relatively easily make it to the end of each level. A tricky part of the programming changes was that the code placed the random treasures in each room and the number of treasures was set based on a 20x20 maze, so he had to scale the number of treasures appropriately, and there was some tricky math in the code.



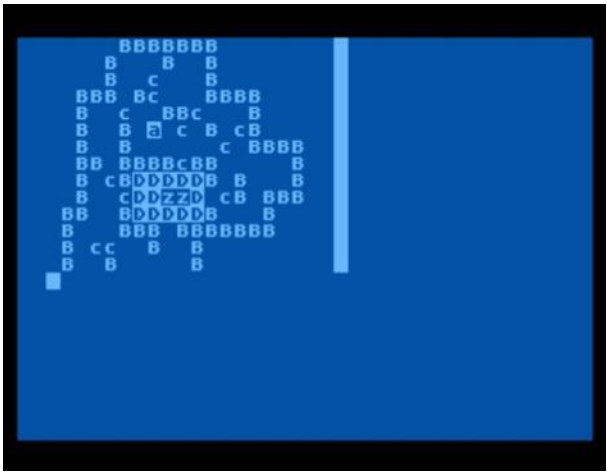

More information about the technical aspects of this game can be read in the original accompanying article sent to computer magazines, A/W/A Game Write-Ups section of this document.

# SOKO-BAN (1988)

http://www.atarimania.com/game-atari-400-800-xl-xe-soko-ban_4784.html





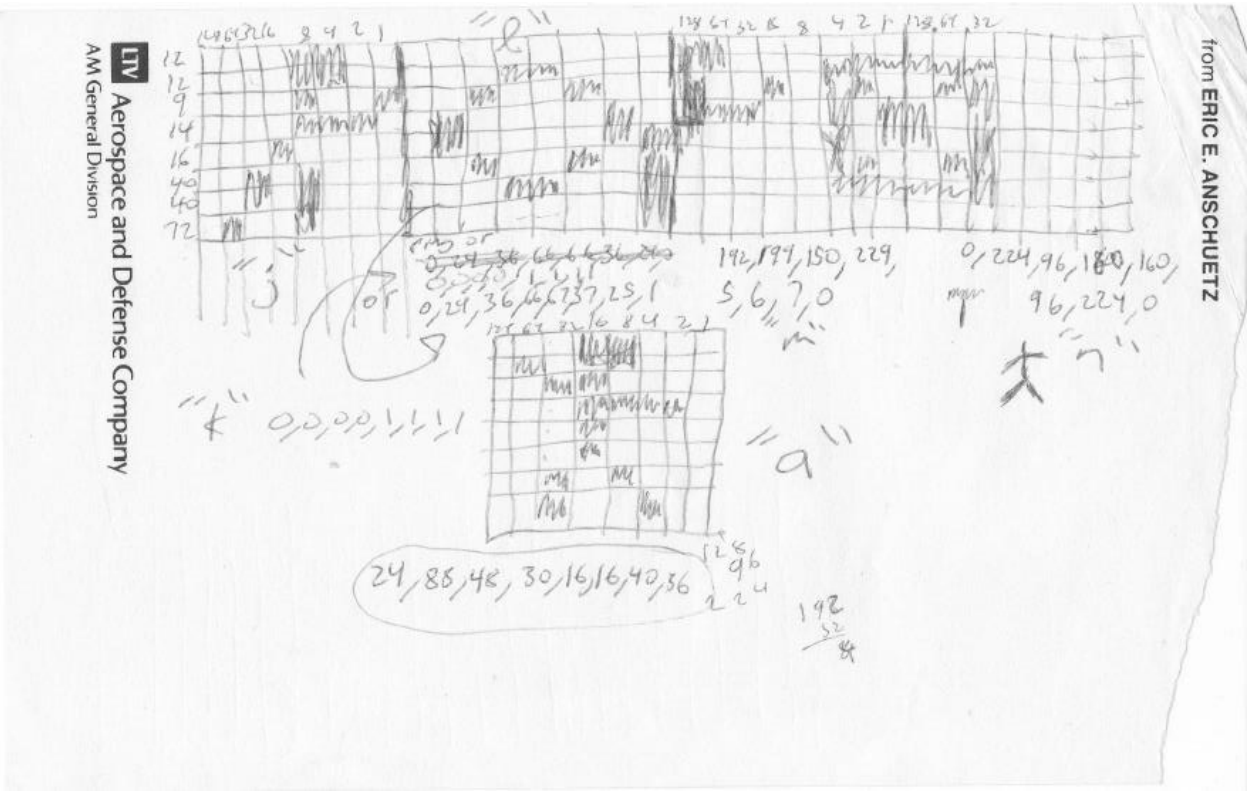
**Game Play**

Sokoban is an addictive puzzle game that was popular to play on PCs about the same time as Tetris came out. After entering your initials and selecting one of 55 levels, your objective is to move several blocks into the landing area. To move a block, the player positions the character behind a block and pushes the joystick in the direction he wants to move the block. The difficult part is getting the player positioned behind each block, so the player needs to be careful not to push blocks against walls that he cannot maneuver behind. Each level is a different puzzle with a different room configuration that must be solved differently. At the bottom of the screen, there is a level indicator. Above that, there is a timer, plus indicators that keep track of the number of moves you make and the number of times that the blocks are moved.

**Recollections**

This game was written entirely by Eric Anschuetz. The program employs character set redefinition and no Player/Missile graphics. The game makes use of the Atari's built in real-time clock to keep track of the time it takes to complete each level. The game also has a separate level editor program that was used to create the various levels. Each level is stored on the disk drive, and loaded into the game as needed. Because of the limit of 64 files on the Atari DOS 2.0 filesystem, there was a limit to the number of levels in the game.

Eric had played PC Sokobon at work during "lunch breaks" with some co-workers. This was in 1988 or 1989 when there were very few PCs. Several engineers worked together to solve the puzzles. They ended up solving every one of them. The Atari game uses the same exact puzzles (plus a few extra levels that were built at Eric's work using the original PC game's level editor).




The game plays exactly like the PC original, but with slightly worse graphics. One of the screens on the original was impossible to solve (as Eric and his colleagues figured out), so it was modified at Eric's work and also in this game. This is the last game that the A/W/A Team ever wrote!

# SPACE ASSAILANTS (1986)

http://www.atarimania.com/game-atari-400-800-xl-xe-space-assailants_31442.html



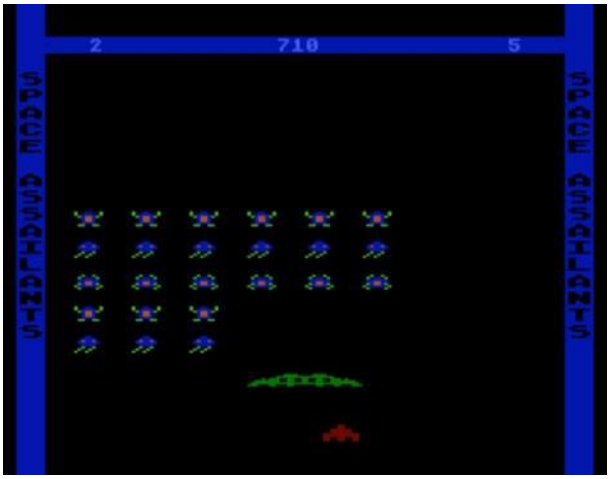
















**Game Play**

Space Assailants is an obvious Space Invaders clone, with some slightly different twists. Instead of avoiding shots fired by the invaders, the player must dodge a starfield that constantly scrolls vertically from top-to-bottom and repeats. The game starts with only one row of invaders. After you clear the first set, the next set requires the player to shoot down two rows of invaders. On the next set, a third row of invaders arrives. The pattern continues until there are six rows of 6 invaders once you reach the 6th level, but the characters in rows 4-6 are a repeat of the same characters used in rows 1-3.

The invaders move across the playfield until they hit one end of the screen, then they drop down a level, reverse direction, and head the other way. The invaders are arranged in a hovering position above their spaceship. Once the spaceship lands at the bottom of the screen, or the player loses all three lives, the game is over. The status banner at the top of the screen indicates the number of lives, the score, and the level. Once the game is over, an end-of-game screen shows you this information and lets you start anew.

The following description of the gameplay was written by the A/W/A Team to be submitted to a computer magazine along with the game.

> *In the game Space Assailants, you are earth's lone defender in a battle which takes place in the midst of a treacherous meteor shower. You must protect the earth from an invading fleet of evil aliens, while avoiding the hurtling meteors. Your main concern is keeping the mothership, which leads the enemy armada, from landing. Although your weaponry cannot breach the powerful shields of the mothership, you can prevent its landing by destroying the accompanying fleet of alien vessels. The fleet drops each time it reaches a side of the playfield in its back and forth motion. After the fleet has dropped ten times, the mothership is considered to have landed, and your game is over. If you are able to destroy the fleet before the mothership lands, you will move on to the next round.*
>
> *Rounds one through six consist of an alien fleet which increases in size from six to thirty-six at the rate of six per round. Rounds seven through twelve are exactly like one through six, but the aliens' speed is doubled. During rounds thirteen through eighteen, the aliens move slowly again, but the meteors travel at double speed. In rounds nineteen through twenty-four, both the aliens and meteors move doubly fast. Sets after this remain at the same level of difficulty as sets nineteen through twenty-four.*
>
> *If you are hit by a meteor, you lose one of three lives and must start your current round over. You score ten points for each alien vessel destroyed. Your score is located at the top and middle of the screen. To its left are lives remaining, and to its right is the round number.*
>
> *You may only fire one missile at a time, and you must wait until the missile either hits an alien or clears the screen to fire again. As the rounds progress,*

*and there are more aliens to contend with, you will find that the best strategy consists of destroying the end columns of assailants, thus lengthening the amount of time between drops of the fleet. Destroying the bottom row of aliens will not delay the mothership's landing.*

**Recollections**

Space Assailants was primarily written by John Weisgerber and really pushed the envelope with what could be done with Atari Basic. The scrolling stars are actually in a different display list than the invaders, and the two screens are alternated very rapidly (every 1/60$^{th}$ of a second) so the eyes perceive them to be on the screen at the same time. In the screenshots above, only either the invaders or the stars are visible, but the images are combined for the player's experience. There is a flickering effect (like the one criticized so heavily in the Atari 2600 Pac-Man), but the flickering effect wasn't too dramatic on old raster-scan televisions.

The banners on the left and right edge of the screen are Player/Missile graphics. The ship and its shot are also Player/Missile graphics. The stars are low resolution graphics pixels. The invaders are re-defined characters in multi-color mode, and they animate with every move in two different poses. The game makes use of the typical droning vibrato background sound effect used in may A/W/A games to take the place of a background melody.

Eric originally wrote an assembly routine to be run during the vertical blank interrupt that would smoothly move the ship left and write based on joystick controls. It worked perfectly except for one problem: John's technique of allowing for interleaving the two graphics modes disabled the vertical blank interrupt, so you couldn't do both! The game as it stands uses BASIC to move the space ship and so it is not as smooth as it would have been in assembly language.

The technique used in this game to interleave two graphics display modes was definitely not used in any other BASIC game or any other game that the A/W/A Team is aware of!

More information about the technical aspects of this game can be read in the original accompanying article sent to computer magazines, A/W/A Game Write-Ups section of this document.

# YAHTZEE (1987)

http://www.atarimania.com/game-atari-400-800-xl-xe-yahtzee_31440.html




**Game Play**

Yahtzee is played like the standard dice game and has the same simple rules. The game can be played by 1 or 2 players. Once the five dice are rolled, the player has the option to re-roll any of the dice to improve his lot. The player then selects which category toward which to apply the dice combinations. There are several categories that must be filled. Of course, rolling all dice with the same number provides a Yahtzee. The player attempts to get the highest score possible for the Upper Total of rolls, which consists of getting 1-of-a-kind rolls of the 6 different dice face possibilities. The Lower Total consists of combinations that are similar to a poker hand. At the end of the game the Upper Total and Lower Total are added together to provide the final score.

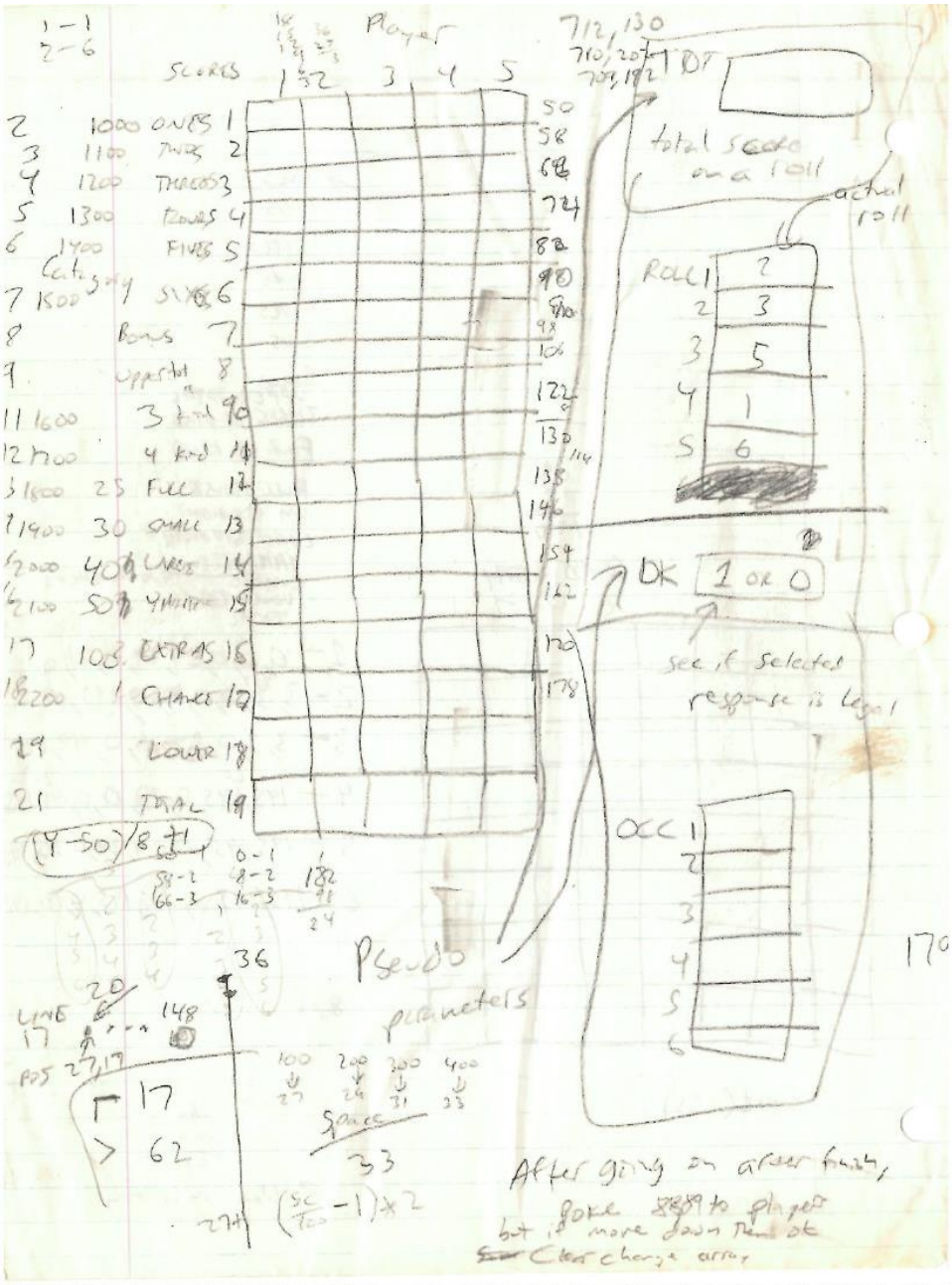
**Recollections**

This is a very simple game, which is only text based. The dice and pointers to the dice and options are Player/Missile graphics, but everything else is rendered in BASIC Graphics 0 mode, which is 40

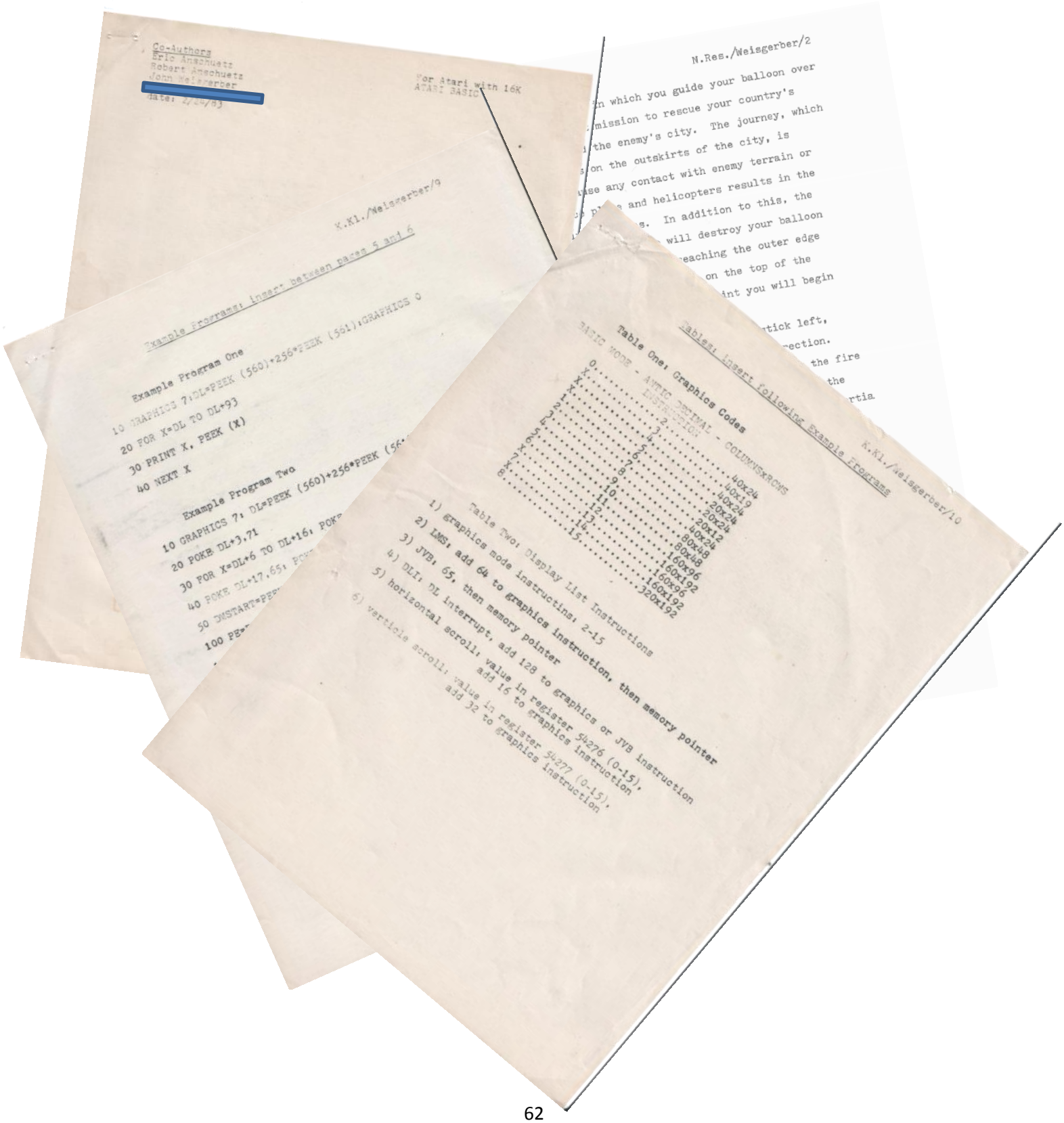
characters x 24 rows of text.  A simple graphic representation of “Professor Yahtzee” is provided using standard Atari character graphics.

Eric wrote this game by himself in his senior year of college (1987).  The dice are made up of a Player/Missile graphic, and actually uses a character string to hold the graphics.  Character Strings in ATARI BASIC had several functions for moving parts of strings around, and these functions were done in very fast Assembly language.  Eric took advantage of these string manipulation functions to quickly change the characters in the dice.

This was the second to last game ever written, just before Sokoban.



# A/W/A GAME WRITE-UPS



# ALIEN ASSAULT



Eric Anschuetz
Robert Anschuetz
John Weisgerber
[redacted]
date: 10/28/83

for Atari with 16K
ATARI BASIC

ALIEN ASSAULT

You have been stationed on the moon in the year 2136. The reason? To protect solar collectors from possible destruction. You will do your protecting from within a super-advanced, armed battle tank. The tank is made of a very strong alloy that cannot be destroyed. Think you are up to the task?

The moon was discovered to have a vast resourse of area available ideal for the positioning of solar collectors. These collectors provide much of the earth's energy and are manditory for the survival of humanity. Your mission: protect the moon, and more specifically, the solar collectors from the ongoing ALIEN ASSAULT.

You begin your mission in the eeriness of night. Intellegent radar indicates that you will have to continue your frenzious attack on the alien bombs on through the spectacular lunar sunrise, onto midday, through the next sunset, and onto another day. Before you begin your mission, you received with a supply of 50 units of energy. As you shoot at the alien bombs with your lasers, however, your supply of energy will rapidly begin to decrease at the rate of one unit per each laser fired.

Earth intellegence has revealed that on the first series of attacks, the aliens will send out ten missles aimed to destroy ten of the solar collectors. During each successive onslaught, five additional missles will be sent until the maximum total of 40 is reached. In addition to more enemy bombs to worry about, on every other wave of bombings, the amount of solar collectors that the aliens have to destroy is decremented by one. This is due to the fact that the solar technicians are too busy fleeing back to earth to have time to make the necessary repairs. Malfunctions do happen occasionally even in this advanced age.

As each alien missle is launched by the alien armada toward the solar collectors, the indicator in the lower middle of the display will indicate whether the missle is the the west(left) or to the east(right) of your tank. To shoot a laser at one of the alien missles, position your crosshairs on top of the missle and press the fire button. A laser will be sent from your tank toward the spot in space directly in line with where the crosshairs are aimed.

If you manage to line the crosshairs directly on top of the alien missle, the missle will be destroyed. Thenumber of missles remaining to be sent will be decremented by one. If you should miss the missle, one of two things may happen. First, the missle may hit a solar collector. If this should happen more than the amount of times allowable for each ambush, your game will be over. Second, the missle may crash harmlessly on the surface of the moon, in which case nothing happens. At the end of each set of ambushes, the color of thebackground will change to indicate the rising of setting of the sun. Every few sets, the missles will speed a bit in their decent.

ALIEN ASSAULT is a difficult game which requires manual dexterity as well as a strategy. Here are some hints. Always keep an eye on the indicator in the middle of the screen. This will tell you which direction to begin moving in. Always begin moving immediately. This will help you get to the enemy faster.

Keep a mental note of where you are in relationship to the entire playing field. If you find you cannot travel any more in the direction in which you wish, you have reached the outside

boundaries and should change directions immediately. As you become more advanced in ALIEN ASSAULT, you will have a feel for when you have begun to travel too far in a particular direction. Sacrifice a solar collector if you feel you are drifting a long way away from the center of the playfield. This may mean many collectors saved in the long run.

Keep an eye out for the energy remaining. This means take carefully placed shots instead of machine-gun style. If you do shoot machine-gun style, you will find that your energy supply will quickly run out and you will be forced to watch helplessly as your last remaining solar collectors are destroyed.

Also watch the number of collectors destroyed in the lower right-hand corner of the screen. Always compare this to the number of collectors that are permitted to be destroyed in the particular ambush. Remember, this number starts at ten and is decreased by one on every other wave.

One final suggestion is to always keep your crosshairs positioned near the bottom. This will give you that all-important last shot. This may prove to be the difference in lasting another ambush or not.

The rest of the article concerns itself with the technical aspects of ALIEN ASSAULT. It is not necessary for the reader to proceed any further, but I recommend that anybody serious in learning to program fast-action games read on. ALIEN ASSAULT features a dual-scrolling, colorful, high-resolution playfield. The playfield is composed of an Antic 5 display and a custom text window. Where there are no graphics on scan-lines in ALIEN ASSAULT, are literally no graphics. Blank scan-lines have been inserted to maximize speed and minimize memory consumption. A

blank line does not require any memory. There is no sense in defining a graphics mode on a line which will have no display on it anyway. There is also no sense in scrolling a line with no graphics on it. This wastes time and, consequentally, speed. There is almost nothing as important as speed when it comes to writing an arcade-style game.

The text window is located below the Antic 5 lines. Since this window is not located exactly where a normal text window would be, a POKE to locations 656 and 657 is necessary before printing to the text window. These lines tell where in text window memory to place the cursor. The necessary information is then printed.

A dual-scrolling background is the only method that can be used to show true perspective. This type of scrolling has been used in the arcade marvels MOON PATROL and JUNGLE HUNT. In ALIEN ASSAULT, the mountains in the background seem much farther away than the solar collectors in the foreground. This is due to the fact that not only are they drawn at roughly the same size as the solar collectors, but they also move twice as fast. This creates a sense of depth. The mountains appear to be many times farther away than the solar collectors do.

The method of scrolling was discussed in our previous article NIGHT RESCUE. The method used in ALIEN ASSAULT is similar with the exception that all lines are not scrolled equally. To scroll the mountains twice as fast as the solar collectors, it is necessary only to scroll the mountains on every other time through the main loop. This is very easy to do and is performed in line 280.

Another aspect of ALIEN ASSAULT worth mentioning is the

extensive(and somewhat complecated)Player-Missle movements. Each alien missle must be kept track of even while off of the screen. This is accomplished by the complecated formulas in lines 220 and 290. As the playfield is scrolled either to the left or to the right, the players for the enemy bombs must be kept track of. The reason for this is that it adds professionalism to the program.

The laser from the battle tank also presented problems. It was necessary to draw the laser from the center of the screen downward. The player is not allowed to go beyond 256 scan-lines. Also, it would be senseless for the bottom of the laser to not show up if the crosshairs were moved too high on the screen. A length of 200 was chosen for the player because this would eliminate both of those dreadful possibilities. The player cannot travel below 256 and the bottom never goes above the bottom of the screen. As the crosshairs are now limited, the laser always is completely visible but does not cause the machine to error because the player is extended too far below the screen.

Finally, if you would like a copy of the game but would not like to type it in, you may send a self-addressed, stamped mailer, a blank cassette, and $3.00 to Robert Anschuetz, [redacted] Be sure to specify which program you wish.

```
100 GRAPHICS 1:SETCOLOR 2,0,0:POKE 16,64:POKE 53774,64:DL=PEEK(560)+
256*PEEK(561):POKE DL+14,23:POKE DL+18,2
110 POKE 54276,4:SETCOLOR 0,8,4:POSITION 5,9:? #6;"press start to
begin"
120 POSITION 9,13:? #6;"ANSCHUETZ/WEISGERBER/ANSCHUETZ"
130 IF PEEK(53279)<>6 THEN 130
140 GRAPHICS 1:POKE 559,0:POKE 16,64:POKE 53774,64:PM=12288:OP=
10:SETCOLOR 0,14,2:SETCOLOR 1,13,10:SETCOLOR 2,12,6
150 POKE 755,0:FOR A=12800 TO 12834:READ D:POKE A,D:NEXT A:DATA 112
160 DATA 112,112,112,112,112,112,112,112,112,112,112,112,69,0,58,
66,35,51,69,0,59,66,35,51,2,2,2,2,66,0,0,2,2,2
170 POKE 560,0:POKE 561,50:POKE 12830,PEEK(660):POKE 12831,PEEK(661)
180 GOTO 450
190 ML=INT(RND(0)*2)+1:MX=INT(RND(0)*784)+120:MY=10:OS=0:NM=NM-1:
IF NM<0 THEN POKE 77,0:GOTO 410
200 IF MX<(PE+20)*4 THEN POKE 656,0:POKE 657,18:? "WEST"
210 IF MX>(PE+20)*4 THEN POKE 656,0:POKE 657,18:? "EAST"
220 POKE PLY+ML,MY:MY=MY+2+(MIAD*0.25):IF MY>130 THEN POKE PLX+
ML,0:GOTO 350
230 IF STICK(0)=7 THEN PE=PE+2*(PE<216)
240 IF STICK(0)=11 THEN PE=PE-2*(PE>0)
250 IF STICK(0)=14 THEN CHY=CHY-(CHY>40)*4:POKE PLY,CHY
260 IF STICK(0)=13 THEN CHY=CHY+(CHY<128)*4:POKE PLY,CHY
270 IF STRIG(0)=0 THEN POKE 53278,0:POKE PLY+3,CHY+1:POKE PLX+3,
124:EA=EA-1:POKE 656,1:POKE 657,7:GOTO 300
280 POKE 12814,PE/2:POKE 12820,PE
290 SMX=MX-PE*4+40+OS:OS=OS+(ML=1)*-2+(ML=2)*2:POKE PLX+ML,(SMX>48)*
(SMX<202)*SMX+(MIAD*0.25):GOTO 220
300 ? EA;" ":SOUND 0,100,10,8:SOUND 0,102,10,8:SOUND 0,105,10,8:
SOUND 0,110,10,8:SOUND 0,0,0,0
310 IF PEEK(53260) THEN SOUND 0,200,2,10:FOR X=0 TO 20:NEXT X:SOUND
0,0,0,0:POKE PLX+ML,0:SC=SC+100
320 POKE PLX+3,0:POKE PLY+3,250:POKE 656,1:POKE 657,18:? SC;:IF
PEEK(53260) THEN 190
330 IF EA<1 THEN 760
340 GOTO 280
```

```
350 HP=INT((MX+OS)/4-0.5)+256*59:IF PEEK(HP)=13 THEN POKE HP,26:
SOUND 0,255,10,10:FOR X=1 TO 25:NEXT X
360 IF PEEK(HP)=26 THEN SOUND 0,0,0,0:GOTO 190
370 IF PEEK(HP)=11 THEN POKE HP,14:POKE HP+1,15:MSL=MSL+1:GOTO 390
380 IF PEEK(HP)=12 THEN POKE HP,15:POKE HP-1,14:MSL=MSL+1
390 IF MSL>OP THEN 760
400 POKE 656,1:POKE 657,31:? MSL;:SOUND 0,150,4,10:FOR X=1 TO 50:
NEXT X:SOUND 0,0,0,0:GOTO 190
410 FOR LN=1 TO 3:SOUND LN,0,0,0:NEXT LN:GH=GH+1:MIAD=MIAD+1:PE=108:
NM=MIAD*5+10:OP=OP-0.5:IF NM>40 THEN NM=40
420 IF GH>15 THEN GH=0
430 SC=SC+(NS-MSL)*10:POKE 656,1:POKE 657,18:? SC;:EA=50:MSL=0:
FOR JW=1 TO 300:NEXT JW:POKE 559,0
440 POKE 656,1:POKE 657,7:? "50":POKE 656,1:POKE 657,31:? "O":GOTO 590
450 RESTORE 630:FOR I=1536 to 1706:READ A:POKE I,A:NEXT I:FOR I=1774
TO 1787:POKE I,0:NEXT I
460 FOR I=PM+1023 TO PM+2047:POKE I,0:NEXT I:FOR I=PM+1025 TO PM+1031:
READ A:POKE I,A:NEXT I
470 FOR I=PM+1281 TO PM+1296:READ A:POKE I,A:NEXT I:FOR I=PM+1793 TO
PM+1797:READ A:POKE I,A:NEXT I
480 FOR I=PM+1537 TO PM+1552:READ A:POKE I,A:NEXT I:FOR I=PM+1799 TO
PM+1992:POKE I,8:NEXT I
490 POKE 704,132:POKE 705,70:POKE 706,70:POKE 707,12:PLX=53248:PLY=
1780:PLL=1784:POKE 623,1
500 POKE 1788,PM/256+4:POKE 53277,2:POKE 54279,PM/256:Q=USR(1696):
POKE PLL,7:POKE PLL+1,16:POKE PLL+2,16
510 POKE PLL+3,200:POKE 53257,2:POKE 53258,2:POKE 53259,2
520 POKE PLX,124:POKE PLY,80:CHY=80:POKE PLY+3,221
530 PE=108:EA=50:NM=10:CH=PM:FOR I=0 TO 511:POKE CH+I,PEEK(57344+I):
NEXT I
540 FOR X=8 TO 127:READ A:POKE CH+X,A:NEXT X:POKE 756,CH/256:X=14848
550 POKE X,INT(RND(0)*2)+1:X=X+1:PN=INT(RND(0)*5)+1:FOR X=X TO X+PN:
POKE X,INT(RND(0)*6)+5:NEXT X
560 POKE X,INT(RND(0)*2)+3:IF X<14995 THEN X=X+1:GOTO 550
570 POKE 656,1:POKE 657,7:? "50          0             0"
```

```
580 POKE 656,2:POKE 657,5:? "ENERGY        SCORE    DESTROYED"
590 FOR X=15104 TO 15359:IF INT(RND(0)*5)+1=1 THEN POKE X,11:X=X+1:
POKE X,12:NS=NS+1:NEXT X:GOTO 610
600 POKE X,13:NEXT X
610 POKE 12814,54:POKE 12820,108:POKE 559,62:SOUND 1,199,4,3:SOUND
2,200,1,4:SOUND 3,201,2,3
620 SETCOLOR 4,GH*2,8*(GH<>0):GOTO 190
630 DATA 162,3,189,244,6,240,89,56,221,240,6,240,83,141,254,6,106,
141,255,6,142,253,6,24,169,0,109,253,6,24,109
640 DATA 252,6,133,204,133,206,189,240,6,133,203,173,254,6,133,205,
189,248,6,170,232,46,255,6,144,16,168,177
650 DATA 203,145,205,169,0,145,203,136,202,208,244,76,87,6,160,0,177,
203,145,205,169,0,145,203,200,202,208,244
660 DATA 174,253,6,173,254,6,157,240,6,189,236,6,240,48,133,203,24,
138,141,253,6,109,235,6,133,204,24,173,253
670 DATA 6,109,252,6,133,206,189,240,6,133,205,189,248,6,170,160,0,
177,203,145,205,200,202,208,248,174,253,6,169
680 DATA 0,157,236,6,202,48,3,76,2,6,76,98,228,0,0,104,169,7,162,6,
160,0,32,92,228,96
690 DATA 28,8,65,99,65,8,28,24,24,24,24,28,28,63,63,63,63,120,120,
224,224,192,192,34,20,8,28,34
700 DATA 24,24,24,24,56,56,252,252,252,252,30,30,7,7,3,3
710 DATA 0,0,0,0,1,5,23,127,0,0,1,5,5,61,63,255,0,0,0,0,64,80,208,
255,0,0,0,64,80,112,245,255
720 DATA 0,2,5,85,87,127,255,255,160,168,88,85,85,223,255,255,0,128,
164,85,93,127,255,255
730 DATA 0,0,160,149,85,85,253,255,32,164,84,85,87,127,255,255,2,10,
137,85,223,255,255,255
740 DATA 255,245,212,80,84,213,251,239,255,95,87,23,95,111,187,254,
255,255,255,255,255,255,255,255
750 DATA 255,255,255,255,209,68,209,245,255,255,255,127,31,71,31,255,
255,255,255,255,115,69,81,247
760 FOR RA=0 TO 3:POKE PLX+RA,0:SOUND RA,0,0,0,0:NEXT RA:GRAPHICS 17:
POKE 16,64:POKE 53774,64
770 POSITION 5,10:? #6;"game over":POSITION 4,13:? #6;"score: ";SC:
GOTO 770
```

# KOOKY DIVER

Co-Authors
Eric Anschuetz
Robert Anschuetz
John Weisgerber
[redacted]
date: 10/25/83

For Atari with 16K
ATARI BASIC

Kooky Diver

Yes, he's back and he's kockier than ever. But this time, instead of climbing another building, Kooky has taken the opposite extreme and has begun a search for lost treasure. On his dives, he will encounter black pearls, treasure chests, pistols, vases, steering wheels, and many other treasures. But what Kooky doesn't know as he begins his expedition is that not only are there a dangerous shark and a school of man-eating piranha to block his path, but that a giant clam has swollowed all of the treasure! You, as Kooky, will have to dive for the various treasures and attempt to bring them up to the salvage boat. To move Kooky in any particular direction, just move the joystick in the corresponding direction. To tred water, push the joystick in any of the diagonal directions. You must time it just right so that the clam's mouth is open when you touch the treasure in order to have a chance to make a successful salvage. If the mouth is open and you are touching the treasure, merely press the fire button on the joystick and you will have picked up the treasure. Of course, if you touch any part of the clam when its mouth is closed, you will lose a life. After you have a treasure in your possession, you will use up your limited supply of oxygen rather quickly. Luckily, there are three coral reefs for you to deposite your treasure on while surfacing for more oxygen. Provided that your current oxygen level is below 50 units, you will completely fill your tanks every time you bring your head out of water. To drop off a treasure on any reef, all you must do is make sure that Kooky is only touching the yellow portion of the reef. At this point, the treasure will automatically be set there with no further actions required. To pick up a treasure again, simply press the fire button while touching the treasure as before. Of course, any collision with the piranha or the shark will cause instant death and send Kooky plummetting toward the ocean floor. After three deaths, your game will end and the high score will be shown. At this point, you can press the "START" button on the computer to start a new game with three new divers. You may end up doing this quite a few times at first because after all, nobody ever said that diving was easy!

The ultimate goal of the game, however, is to deposite your treasure in the boat that is moving across the top of the screen. To do this, just touch the boat and your treasure will appear inside. At this point, a new treasure will appear inside the clam, and you can now try to get this treasure to the boat. Now that you know how to play the game, you can either type in the program now and begin playing immediately, or you can continue to read the rest of this article to learn just how KOOKY DIVER was written.

This program uses the machine language vertical player movement routine from the article by Tom Sak (August, 1982) and combines it with two display list interrupts in order to make an attractive, fast moving arcade game. The first display list interrupt (Figure 1) occurs between the last antic mode four line, which is the last line of the actual display field, and the line of regular graphics mode zero, which displays the player's oxygen left, current score, and the number of divers remaining. This interrupt is used to change the character set pointer at memory location 54281 in such a way that the upper part of the screen will continue to display the redefined characters that are used to draw the boat, coral reefs, and the clam, while the lower line of graphics zero will use the standard character set located at ROM location, 57344. During this interrupt, the color register at location 53272 is changed to produce a color of light green for the background of the text line, and the display list interrupt pointer is modified to point to the second display list interrupt which occurs on the next line.

This second interrupt (Figure 2) is really the key to making the program look graphically appealing. In order to make the school of piranha and the shark move at the same speed without the use of this interrupt, BASIC alone would have to be used to move both of them. This would both slow down the rest of the program and cause a choppy jump of six pixels, or more, each time the fish moved. In order to provide smooth motion, machine language had to be used. As a beginning machine language programmer myself, I was a little hesitant to try to code this routine, but I decided to give it a try, and as you

can see, I successfully managed to do so. Hopefully with the comments that I have provided, you too can begin to write some short routines in machine language without even the use of an ASSEMBLER program, as I did. As in the first routine, it was necessary to modify the display list interrupt vector to point to the other interrupt, and it was also necessary to store any value in memory location 54282 in order to eliminate the scan line jump that is often seen in many display list interrupt routines.

As you can see, in the program many pictures are used to draw both Kooky and the treasures that he must pick up. For this reason, at the appropriate times, the values of locations PDR and PDR+1 are modified to point to the proper drawings needed. This technique was successfully used to make Kooky animated, as well as making it possible to have eight different possible treasures to get.

Again, to make the program look much better, the salvage boat at the top scrolls across the screen, and what's more, its movement is contained entirely in just one line: 220. The scrolling routine is set up in lines 390 through 430 and is not even looked at by the program again. As it is plain to see, the little work required in setting up a scrolling routine is more than made up for by the spectacular visual effects of the scroll.

One final comment is in order: When I first completed this program, I came up several hundred bytes short of fitting it into 16K of memory. For this reason, I used the variable, C0, and set it equal to zero at the beginning of the program. Then I proceeded to change all RND(0) statements to RND(C0), all SOUND 0,0,0,0 statements to SOUND C0,C0,C0,C0 , and all variables which were once set to zero were changed to be initialized to C0. This technique alone saved enough memory to fit the entire program into 16K, and could have been used even more extensively to save up to a thousand bytes of memory, or more, on any given program. Of course, I wouldn't have been

in any position at all to start thinking of saving a couple hundred bytes of memory if I hadn't used a redefined character set in the first place to save about seven thousand bytes by using an antic mode four screen instead of an antic mode fourteen screen. This also made is much easier to redraw the clam every time it has to open and close its mouth. Just a few PRINT statements take the place of hundreds of PLOT and DRAWTO statements, so if you haven't started using these redefined character sets yet, I urge you to at least try and see just how easy they can be to use. Well, that just about covers all of the technical aspects of the game, while the rest of the program is written in a straight-forward manner that should be fairly easy to understand.

If you don't have the time to type in KOOKY DIVER yourself, you can get a copy by sending $3.00 and a blank cassette in a self-addressed, stamped mailer to Eric Anschuetz, 101 E. Forest, Ypsilanti, Mi. 48197. Be sure to include the name of the program that you want.

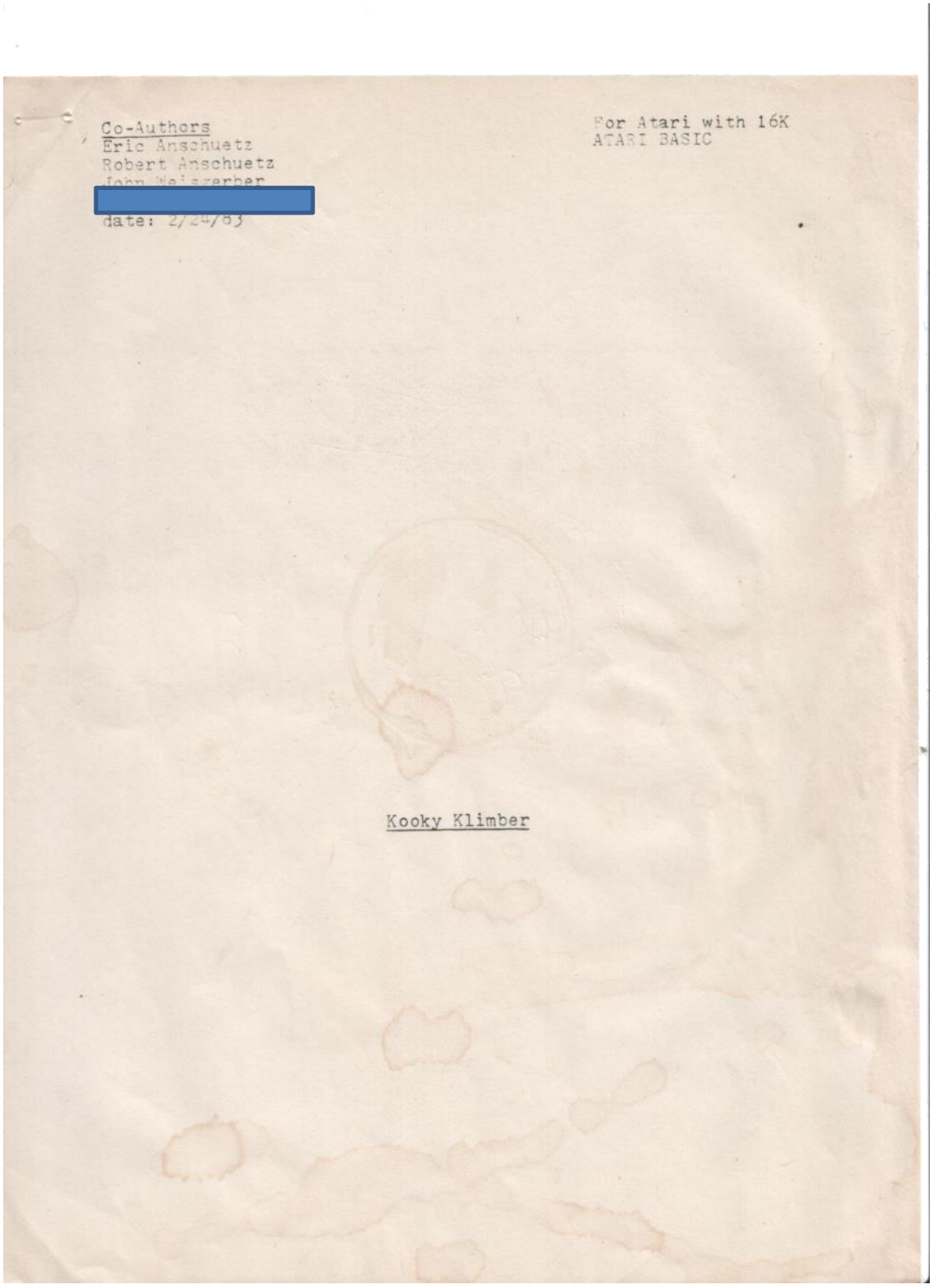
Figure 1

| CODE | Instruction | BASIC Equivalent | Comment |
|---|---|---|---|
| 72 | PHA | | Save the current value of A |
| 169,224 | LDA #224 | A=224 | Store 224 in A |
| 141,10,212 | STA 54282 | POKE 54282,A | Store A in 54282 |
| 141,9,212 | STA 54281 | POKE 54281,A | Store A in 54281 |
| 169,212 | LDA #212 | A=212 | Store 212 in A |
| 141,24,208 | STA 53272 | POKE 53272,A | Store A in 53272 |
| 169,195 | LDA #195 | A=195 | Store 195 in A |
| 141,0,2 | STA 512 | POKE 512,A | Store A in 512 |
| 104 | PLA | | Restore old value of A |
| 64 | RTI | RETURN | Back to BASIC |

Figure 2

| CODE | Instruction | BASIC Equivalent | Comment |
|---|---|---|---|
| 72 | PHA | | Save current value of A |
| 169,250 | LDA #250 | A=250 | Store 250 into A |
| 197,208 | CMP 208 | IF PEEK(208)<>250 | See if PEEK(208)<>250 |
| 208,7 | BNE MOVE | THEN GOTO MOVE | If greater then go to MOVE |
| 169,0 | LDA #0 | A=0 | Store 0 into A |
| 133,208 | STA 208 | POKE 208,A | Store A into 208 |
| 24 | ~~CLC~~ | GOTO | Clear Carry flag and |
| 144,15 | BEQ ~~BCC~~ QUIT | QUIT | force a branch to QUIT |
| 230,208 | MOVE INC 208 | POKE 208,PEEK(208)+1 | Increment contents of 208 |
| 165,208 | LDA 208 | A=PEEK(208) | Store contents of 208 in A |
| 141,2,208 | STA 53250 | POKE 53250,A | Store A into 53250 |
| 169,255 | LDA #255 | A=255 | Store 255 into A |
| 56 | SEC | | Set Carry flag |
| 229,208 | SBC 208 | A=A-PEEK(208) | Subtract contents of 208 |
| 141,3,208 | STA 53251 | POKE 53251,A | Store A into 53251 |
| 141,10,212 | QUIT STA 54282 | POKE 54282,A | Store A into 54282 |
| 169,174 | LDA #174 | A=174 | Store 174 into A |
| 141,0,2 | STA 512 | POKE 512,A | Store A into 512 |
| 104 | PLA | | Restore old value of A |
| 64 | RTI | RETURN | Back to BASIC |

# KOOKY KLIMBER



Co-Authors
Eric Anschuetz
Robert Anschuetz
John Weisgerber
[redacted]
date: 2/24/83

For Atari with 16K
ATARI BASIC

Kooky Klimber

After reading the two articles on P/M graphics by Tom Sak and Sid Meier (February and August, 1982),we decided that it would be easy to adopt their animation techniques toward an arcade game. After combining the animation with rough scrolling, we came up with Kooky Klimber, a fast action arcade style game based loosely on a popular coin-op game.

In this game, you are Kooky Klimber. You have an incurable urge to climb tall buildings. After a period of initialization, you find yourself poised at the bottom of the first (grey) building. To ascend the building you must pull the joystick back and then push it forward. After this, your klimber will automatically climb to the next story. To move left and right simply push the joystick in the corresponding direction. Sound easy? Well, as you will soon discover, it's not!

In the course of your climb, the people inhabiting the building will be opening and closing the windows. If one of these windows should close on your hands, you will find yourself falling head first towards the bottom of the building. A second nemisis directd towards your klimber is the falling objects. Clumsy residents above have a deadly habit of knocking flowerpots off of their windowsills. These flowerpots can do lethal harm to your klimber. If they should touch any part of Kooky Klimber, he will lose his grip on the building and fall to a quick death.

Another clutsy group of people above you are the construction workers. They are continually dropping steel girders which flip through the air while they fall. As is the case with the flowerpots, any contact between your klimber and the girder will cause you to slip and fall.

Your game starts with three Kooky Klimbers. A bonus klimber is earned after achieving 10,000 points. Once you reach the top of any building, you will be transported to the bottom of another building with a different color and harder skill level.

The sophisticated P/M graphics used in this game were actually quite easy to create using the machine language subroutine by Mr. Sak and Mr. Meier. Four images are used for the klimber and the girder, while the flowerpot uses only one. Every time through the main loop of the program, the variable DRAW is incremented in such a way that the following time through the loop, the next drawing will appear. For example, on the first time through the loop, the player appears to have both hands on the ledge, while one hand appears higher then the other on the next time through. These changing drawings continue until a complete cycle of three drawings is made, and then it starts all over. Also, at the beginning of the first loop of the program, a random choice of which object is to be dropped is made. If the flowerpot is dropped, the movement is achieved by simply incrementing its Y-position each time. If, however, the girder is dropped, the movement is more complicated, because the girder uses animation much like the klimber. In either case, checks are made to see if the object hits the klimber or goes off the bottom of the screen. If an object has gone off the bottom of the screen, a new random selection is made to determine which object will fall next.

The only other things necessary to do with the P/M graphics are minimal. For one thing, a check is made to see if the klimber falls off the left or right side of the building. It is also necessary to stop the objects from falling if the klimber

gets too close to the top of the building. After all, it would seem silly, or should we say "kooky", to have girders and flowerpots falling out of mid-air. Finally, aside from checking to see if the klimber collides with the objects (which is done, incidentally, using the player one collision register of 53261), a check is needed to see if a window closes on his hands. This requires a fairly tricky step of overlapping two players. This is necessary because we only want to check for a window closing on his hands, not his feet. Therefore, after the klimber advances to the next floor of the building, a second player, which is defined to look exactly like the klimber's hands, is moved into the position of the hands. This is not visible to the gamer, and is a very effective way of determining if a window has closed on the klimber's hands. Before the klimber's next move, the additional hands are positioned off the visible telivision screen which essentially erases them.

The next part of this article explains, in detail, the workings of a display list, or DL, and briefly discusses the method of rough scrolling used in Kooky Klimber.

If you don't know the details of how a display list works yet, it's time to learn. A display list is a machine language program for antic, Atari's graphics chip. Since there are very few commands for antic, these programs are relatively simple. The DL really only tells antic two important things. First, it tells antic which graphics mode it will display for each line on the screen. Second, it tells antic where it will find the memory to display on these lines. This is an oversimplified explanation, but not much.

Right now you should realize that, at first, it is easier to modify an existing DL than to write your own. For instance, if you want to write a scrolling program in graphics 2 from BASIC, you'll probably want more screen memory than the 240 bytes given to you in graphics 2. In this case you could simply go into graphics 7, then convert the DL which is already there into a graphics 2 DL. This way you get 3840 bytes of display memory which is equivilent to 16 screens of graphics 2. One big advantage of this method is that by using the graphics 7 statement you absolutely protect your screen memory from infringement by the BASIC program. If, however, you want to use every possible byte of memory, you can simply pick the address at which you want to start your DL and write it in at this point. This is the method we used in Kooky Klimber. We used basically a graphics 3 DL written in directly following the space reserved for our P/M graphics. This gave us enough space for a fairly tall building.

Now that you can see the usefulness of modifying a DL, I'll explain how. It's much easier to work with a DL if you have an idea of what one looks like, so, if you have an Atari handy, type in example program one. Running this program locates the memory address of the graphics 7 DL by going into graphics 7 then checking the values in addresses 560 and 561 which tell antic where to find the DL. The formula used to calculate this memory location is an important one: the first byte plus 256 times the second byte. There are two main types of instructions for antic: one-byte and three-byte. This formula is important because in all three-byte instructions the last two bytes give a memory location using the formula, while the first byte serves to indicate the specific purpose of the instruction.

Now we can go through and examine the DL, instruction by instruction. The first three instuctions in the DL are all one-byte instructions. These tell antic to leave lines blank. The formula for this type of instruction is the number of lines you wish left blank minus one times 16. Since these instructions are 112's on all Atari DL's, they tell antic to leave three sets of eight lines blank right away. These are used so the display won't start above the top of your television screen. After these comes the first three-byte instruction. This is a load memory scan or LMS instruction; it tells antic where to look for the memory it will diplay on the screen. The first byte of this instruction is a combination of the LMS indicator, which is 64, and a graphics 7 instruction, which is 13, for a total of 77. Actually, this instruction only tells antic where to find the memory for this first line of graphics 7, and it is possible to use these instructions to point out memory for each of the lines individually. Fortunately, you don't have to do this, because, for each successive line after the first LMS instruction, antic just uses the memory directly following what it used for the last line. As I said before, the last two bytes of a three-byte instruction are the ones that contain a memory pointer. So, by using our formula of the first byte (actually the second in a three-byte instruction) plus 256 times the second, you can calculate the start of graphics 7 display memory. After the LMS instruction come a lot of 13's. Each of these tells antic to display one line of graphics 7 (a complete list of these instructions is given in table one) just as the first byte of the LMS instruction did, except these don't need the added 64 (LMS indicator) or memory pointers. Directly following the graphics 7 instructions

is another LMS instruction followed by three 2's. This is the set of instructions for the text window, which is like a miniature display list within a display list. Next come the last three bytes of the DL. These are a jump on vertical blank or JVB instruction. This tells antic the location of the start of the DL so it can run through the whole thing again (it does this 60 times per second). The indicator for a JVB instruction is a 65, which is not added to a graphics instruction. Using the memory pointer formula again, you can see for yourself that it does indeed point to the beginning of the DL. The antic instructions I've mentioned so far are enough to allow you to write your own DL, but a complete (as far as I know) list of instructions is given in table two.

Example program two utilizes the techniques I have previously discussed by converting a graphics 7 DL to graphics 2. It also includes a method of rough scrolling in BASIC. The first line of the program puts the screen into graphics 7 and locates the beginning of the DL. The second line begins the 7 to 2 conversion by changing the first byte of the LMS instruction, which had been 77 (64+13) to 71 (64+7, since 7 is the antic code for graphics 2). Line 30 converts 11 more lines to graphics 2 for a total of 12 lines. Line 40 finishes off the conversion by poking in a JVB directly following the graphics 2 instructions. Since I don't know how much memory everyone will use when typing this program in, the numbers for the pointer of the JVB are found by peeking the original graphics 7 JVB. The fact that the rest of the old graphics 7 DL is still there doesn't matter, since antic will never get there, jumping to the top of the DL each time it reaches the new JVB.

Lines 50 through 125 give an example of simple rough scrolling. Now that you know the function of a load memory scan instruction, this method of scrolling should be easy for you to understand. Since the LMS instruction tells antic where to find the first byte of memory it will use to display on the screen, all we have to do is change this instruction and antic will look where we tell it to, making the image on the screen appear to move. In fact, this is what lines 110,120 and 125 do; they change the value of the pointer part of the LMS by 20 (the number of bytes in one line of graphics 2), making the asterisk which has been poked onto the screen seem to move up and down. The variables PE and PE1 keep track of the values in the second and third bytes of the LMS instruction. If you don't understand exactly how a memory pointer works, I'll try to clarify things a little. In a normal two digit number, say 27, there is a tens digit and a ones digit. In order to calculate the value of this number, you could add the ones digit (7) to ten times the tens digit (2). If, however, you get a ones digit greater than nine, you must bump up the tens digit by one and drop the ones digit back down. This is how a memory pointer works, except that instead of ones and tens, you're dealing with ones and 256's: the first byte is the ones and the second byte is the 256's. You can now understand why, in line 120, when PE goes above 255, we must add one to PE1 and subtract 256 from PE.

I hope this article has helped clear up some of the mysteries of the DL for you. The uses of modified DL's are virtually limitless, and with a little experimentation you should be completely comfortable working with one.

Example Programs: insert between pages 5 and 6

Example Program One

```
10 GRAPHICS 7:DL=PEEK (560)+256*PEEK (561):GRAPHICS 0
20 FOR X=DL TO DL+93
30 PRINT X, PEEK (X)
40 NEXT X
```

Example Program Two

```
10 GRAPHICS 7: DL=PEEK (560)+256*PEEK (561)
20 POKE DL+3,71
30 FOR X=DL+6 TO DL+16: POKE X,7: NEXT X
40 POKE DL+17,65: POKE DL+18,PEEK (DL+92): POKE DL+19,PEEK (DL+93)
50 DMSTART=PEEK (DL+4)+256*PEEK (DL+5):POKE DMSTART+130,10
100 PE=PEEK (DL+4): PE1=PEEK (DL+5)
110 IF STICK(0)=14 THEN PE=PE-20: IF PE<0 THEN PE=PE+256: PE1=PE1-1
120 IF STICK(0)=13 THEN PE=PE+20: IF PE>255 THEN PE=PE-256: PE1=PE1+1
125 POKE DL+4,PE: POKE DL+5,PE1: GOTO 110
```

Tables: insert following Example Programs

**Table One: Graphics Codes**

| BASIC MODE | ANTIC DECIMAL INSTRUCTION | COLUMNSxROWS |
|---|---|---|
| 0 | 2 | 40x24 |
| X | 3 | 40x19 |
| X | 4 | 40x24 |
| X | 5 | 20x24 |
| 1 | 6 | 20x24 |
| 2 | 7 | 20x12 |
| 3 | 8 | 40x24 |
| 4 | 9 | 80x48 |
| 5 | 10 | 80x48 |
| 6 | 11 | 160x96 |
| X | 12 | 160x192 |
| 7 | 13 | 160x96 |
| X | 14 | 160x192 |
| 8 | 15 | 320x192 |

Table Two: Display List Instructions

1) graphics mode instructins: 2-15

2) LMS: add 64 to graphics instruction, then memory pointer

3) JVB: 65, then memory pointer

4) DLI: DL interrupt, add 128 to graphics or JVB instruction

5) horizontal scroll: value in register 54276 (0-15), add 16 to graphics instruction

6) verticle scroll: value in register 54277 (0-15), add 32 to graphics instruction

# KOOKY'S QUEST

Co-Authors
Eric Anschuetz
Robert Anschuetz
John Weisgerber
[redacted]
7-8-84

Atari BASIC
24K cassette
32K disk

Kooky's Quest

Kooky's Quest is an action adventure game. You play using the keyboard and a joystick in controller jack one. There are two skill levels: expert and beginner. In a beginner game, you have a map of the dungeon to follow. If you like the challenge of mapping a game, you should play expert, as you can only see the section of the dungeon you are occupying.

Trapped in an ancient dungeon, your only hope lies in descending to the lowest level and retrieving the magic key. By returning to the first level with the key and placing it in the lock located in one of the level-one rooms, you can free yourself from the evil dungeon and win the game. You have ten lives with which to accomplish this task.

In order to move from one level to another, you must obtain a certain number of treasures (rings and chests). You need only one treasure to go from the first level to the second, but you need five in all other cases. Once you have enough treasures to change levels, you simply move to the up or down staircase, designated by a U and a D on the map. If you are playing without the map, you can only identify a staircase by the sound effects for going up and down, because the stairs aren't visible in the view of your immediate vicinity.

Getting treasures is not an easy task. In each room, you will find a treasure and a monster. It is important to get to the treasure before the monster gets it, before you kill the monster, or before the monster kills you, since, in any

one of these cases, that treasure will be lost to you. If you clear all the rooms on any level and don't have enough treasures to move to another level, you will be forced to wander that level forever.

As you enter a room, you are asked to choose which weapon you will use. Listed at the bottom of the screen are the six weapons available to you. Simply press the number of the weapon you want to use. Three of the weapons, the sword, axe, and mace, are swinging weapons. To swing any one of these, you press the fire button, then push the joystick to the left or right. The knife, lightning bolt, and fireball are throwing weapons. The procedure for using one of these is the same as for a swinging weapon, but it will travel across the room until it hits a monster, a treasure, or the wall. Your choice of weapon is very important, since each type of weapon has a different effect on each monster, taking from one to six hits to kill the monster.

There are six varieties of monster you may encounter during your quest. These include vampire bats, giant hobgoblins, wicked skeletons, venomous serpents, decaying mummies, and, worst of all, malevolent spiders which can catch you in their webs and inhabit only the lowest level.

As a variation of Kooky's Quest, instead of attempting to escape the dungeon with as many lives as possible, you can go for points, seeing how many you can accumulate before you lose all your lives. A treasure is worth five points and a

monster is worth ten. To avoid typing in Kooky's Quest, which is fairly long, send a self-addressed, stamped mailer, a blank cassette, and $3.00 to Robert Anschuetz, [redacted] Please be sure to specify Kooky's Quest.

# NIGHT RESCUE

Co-Authors
Eric Anschuetz
Robert Anschuetz
John Weisgerber
[redacted]
date: 5/31/83

For Atari with 16K
ATARI BASIC

Night Rescue

Night Rescue is a game in which you guide your balloon over enemy territory on a secret mission to rescue your country's spies who have infiltrated the enemy's city. The journey, which begins over the mountains on the outskirts of the city, is extremely dangerous because any contact with enemy terrain or the enemy reconnaissance plane and helicopters results in the loss of one of only five balloons. In addition to this, the enemy has planted double-agents who will destroy your balloon if you attempt to rescue them. Upon reaching the outer edge of the city you must make a soft landing on the top of the headquarters of the resistance, at which point you will begin another run above the city.

The balloon is controlled by moving the joystick left, right, up, and down to move in the corresponding direction. The balloon's position may be "fine-tuned" by pressing the fire button while moving the joystick left and right. Moving the balloon up and down is made treacherous by the forces of inertia acting upon it. To land on the resistance's headquarters you must be both scrolled and "fine-tuned" all the way to the right. Scoring in Night Rescue consists of 100 points for each rescued agent plus a bonus for reaching the resistance's headquarters which increases for each new set. At the end of the game you may play again by simply pressing the START key.

The P/M motion used in Night Rescue is from the article by Tom Sak (February, 1982) which lends itself remarkably well to games written in BASIC. The apparent inertia acting on the balloon is created using a variable the value of which is constantly added to the y-position of the balloon. This makes

it necessary for the player to slow down to a stop then speed up again in order to change the direction of his verticle motion.

The horizontal scrolling used in Night Rescue is achieved in much the same way as the verticle scrolling in our previous game, Kooky Klimber. While horizontal scrolling does have a few twists of its own, the method is still fairly simple. If you didn't read the Kooky Klimber article you may or may not know the importance of the Load Memory Scan or LMS instruction as it pertains to scrolling. If you don't, here it is in a nutshell. The LMS is the instruction which tells antic which memory to display for each line of graphics. By shifting the LMS, and thus the memory antic displays on the screen, you can make something on the screen appear to move or scroll. The only other thing you need to know is that the LMS is a three-byte instruction; the first byte indicates the instruction's function, and the last two are a memory pointer. So when the first and second bytes of the memory pointer are mentioned, it actually means the second and third bytes of the LMS. With this knowledge you will be able to follow the rest of this article fairly well.

In Night Rescue each row of scenery or terrain is allotted its own page of memory. One benefit of this is that since the second byte of the pointer in the LMS of each display line actually contains the value of the page on which the displayed memory is located, and this value never changes during the course of the game, checks dealing with the second byte of the pointer become unnecessary. Another reason for this type of memory arrangement is that it avoids an irritating flicker in the screen image which occurs due to the time lapse involved in the BASIC poke function if both values of the LMS pointer must be changed

simultaneously. Since each page of memory is 256 bytes long, and antic modes four and five, which use 40 bytes per line, are used for the graphics, the terrain can be more than six screens long (which it is) without jumping over to another page.

Even though poking the second byte of the LMS pointer is avoided, each line of graphics must still have its own LMS instruction to tell antic which part of memory to display for that particular line. Since each line of graphics is arranged as it will be displayed from the beginning to the end of the page, the same variable can be used to store the value which will be poked into the first byte of each LMS pointer. This helps keep the BASIC program from becoming too slow, but it is still necessary to keep the number of graphics lines down so that there are as few numbers to be poked as possible; poking locations from BASIC takes more time than one might think. In Night Rescue the number of moving lines is limited to four: three of antic five for the normal terrain, and one of antic four for the helicopters. The spaces on the screen (like between the helicopters and building tops) are the result of blank line instructions which consist of the number of lines (up to eight) you wish left blank minus one times sixteen. The use of these instructions eliminates unnecessary memory usage and makes things generally much neater.

The display of score and the player's balloons remaining is simply printed onto a custom text window. The address of the memory used for this window is found for the various amounts of memory in which Night Rescue is likely to be run by peeking the useful locations of 660 and 661. Another interesting capability

of the Atari used in Night Rescue is the option of creating new character sets. By using location 756 to switch between two character sets, the helicopters are made to switch direction each time the player changes the direction of scrolling. The use of antic modes four and five along with P/M graphics results in a high resolution game which is also highly colorful. Although the scrolling used is technically considered rough, one-fourtieth of the screen is a not unsatisfactory jump which also keeps the game from being monotonously slow.

We think you'll agree that we've stretched the game capacity of BASIC nearly to its limits with Night Rescue. With this article and the program itself as guides, you may come up with some interesting uses for horizontal scrolling of your own. As a final note, you may send a self-addressed stamped envelope plus three dollars and a blank cassette to Eric Anschuetz, [redacted] for a copy of Night Rescue. We make this offer because the program has quite a few data statements which you might not want to type in. In any case, don't worry; it won't be long till you see the next Anschuetz/Weisgerber/Anschuetz game.

# OVERFLOW

Co-Authors
Eric Anschuetz
Robert Anschuetz
John Weisgerber
[redacted]
7-8-84

Atari BASIC
16K cassette
24K disk

Overflow

In Overflow, you're a plumber trapped in the central conduit of a network of pipes. Water is advancing relentlessly toward you through the secondary pipes. If the water gets far enough, it will flood the main pipe, drowning you and ending your game. This is a situation you (as a devout plumber) obviously want to avoid. Luckily, you can temporarily reverse the flow of water in a pipe by hurling your plunger down the pipe at the onrushing column of water. Scoring is based on the fact that when the plunger makes contact, in addition to making the water recede, you earn ten points. Controlling the flow of water is simply a matter of moving up and down in the main pipe and throwing your plunger down the pipes closest to the point of overflowing. Unfortunately, even if you manage to gain control of the situation, at five-hundred points the width of the main pipe doubles, making your job all the more difficult. Movement is controlled from a stick in port one. To move up and down in the central pipe, push the stick forward and back. To throw your plunger, push the stick left and right.

The program for Overflow is fairly simple. Players are used for the plumber, plunger, and the central pipe. All vertical player movement is implemented through the use of strings printed at the proper place in memory (by POKEing locations 88 and 89 to the correct addresses). The main graphics mode used is antic-five. The only unique technique used in Overflow is that of using the same memory (basically) for every line of the playfield graphics. Each line has its

own LMS, and the high byte on every one is set to the same page. Moving the water back and forth in the pipes is done by incrementing the low byte on each LMS by either positive or negative two. The increment for each line is held in a one-dimensional array. If a column of water is hit by the plunger, the increment's sign is simply reversed.

```
100 MEM=PEEK(106):DL=(MEM-5)*256:CH=(M
EM-4)*256:P1=MEM-12:P2=MEM-11:P3=(MEM-
9)*256:S=(MEM-7)*256
110 GRAPHICS 2:SETCOLOR 1,3,2:DIM Q(10
),A$(46),B$(16):POKE 755,0:POSITION 6,
4:? #6;"OVERFLOW":GOTO 390
120 POKE 89,P1:POKE 756,MEM-4:POKE 559
,62:POKE 560,0:POKE 561,MEM-5:POKE 532
51,120:POKE 53248,123
130 ? :? :? :? :FOR X=1 TO 10:Q(X)=INT
(RND(0)*2)*4-2:NEXT X:SC=0:POKE 656,0:
POKE 657,16:? "OVERFLOW"
140 POKE 657,4:? "SCORE:0    ";:POKE 6
57,26:? "HIGH:";HC;:YX=33:POKE 88,YX:P
OSITION 0,0:? #6;A$(17,30)
150 QL=DL+1:C=0:FOR X=DL+4 TO DL+31 ST
EP 3:Y=INT(RND(0)*11)+101:Y=Y+(Y/2<>IN
T(Y/2)):POKE X,Y:NEXT X
160 C=C+1:QL=QL+3:POKE QL,PEEK(QL)+Q(C
)
170 POKE 53278,0:IF QL>DL+28 THEN QL=D
L+1:C=0
180 ST=STICK(0):IF ST=14 THEN YX=YX-16
*(YX>33):POKE 88,YX:POSITION 0,0:? #6;
A$(17,46)
190 IF ST=13 THEN YX=YX+16*(YX<177):PO
KE 88,YX+16:POSITION 0,0:? #6;A$(1,30)
200 IF ST=7 AND J=0 THEN PLX=133:PLY=Y
X+3:J=4:POKE 89,P2:POKE 88,YX+3:POSITI
ON 0,0:? #6;B$(1,8):POKE 89,P1
210 IF ST=11 AND J=0 THEN PLX=115:PLY=
YX+3:J=-4:POKE 89,P2:POKE 88,YX+3:POSI
TION 0,0:? #6;B$(9,16):POKE 89,P1
220 IF PEEK(53255)>3 THEN POKE 707,148
:GOTO 350
230 IF J<>0 THEN GOTO 250
240 GOTO 160
250 IF PEEK(53253)>3 THEN 280
260 POKE 53249,PLX:PLX=PLX+J
270 IF PEEK(53253)<4 THEN 300
280 HQ=(PLY-20)/16:IF SGN(Q(HQ))<>SGN(
J) THEN 330
290 SOUND 0,100,10,10:SC=SC+10:Q(HQ)=-
Q(HQ):POKE 657,10:? SC;:GOTO 320
300 IF PLX>200 OR PLX<48 THEN GOTO 330
310 GOTO 160
320 POKE 77,0:IF SC=500 THEN POKE 5325
9,3:POKE 53251,112
330 POKE 53249,0:J=0:POKE 89,P2:POKE 8
8,PLY:POSITION 0,0:? #6;A$(1,8):POKE 8
9,P1:POKE 53278,0
340 SOUND 0,0,0,0:GOTO 160
```

```
350 J=0:POKE 656,3:POKE 657,9:? "PRESS
 <START> TO BEGIN";:IF SC>HC THEN HC=S
C
360 YC=YC+(YC<255)-255*(YC=255):POKE 7
04,YC:IF PEEK(53279)<>6 THEN 360
370 FOR X=DL+4 TO DL+31 STEP 3:POKE X,
108:NEXT X:POKE 89,P2:POKE 88,PLY:POSI
TION 0,0:? #6;A$(1,8):POKE 53251,120
380 POKE 89,P1:POKE 88,YX:POSITION 0,0
:? #6;"    ":POKE 707,50:POKE 53249,0:
POKE 53259,1:GOTO 130
390 FOR X=0 TO 511:POKE X+CH,PEEK(X+57
344):NEXT X:FOR X=8 TO 23:READ A:POKE
X+CH,A:NEXT X
400 FOR X=DL TO DL+2:POKE X,112:NEXT X
:FOR X=DL+3 TO DL+30 STEP 3:POKE X,69:
POKE X+1,108:POKE X+2,S/256:NEXT X
410 POKE DL+33,66:POKE DL+34,PEEK(660)
:POKE DL+35,PEEK(661):FOR X=DL+36 TO D
L+38:POKE X,2:NEXT X:POKE DL+39,65
420 FOR X=P3+32 TO P3+191:POKE X,255:N
EXT X:FOR X=P1*256 TO P3:POKE X,0:NEXT
 X:FOR X=1 TO 46:A$(X,X)=" ":NEXT X
430 FOR X=17 TO 30:READ A:A$(X)=CHR$(A
+32):NEXT X:A$(46,46)=" ":FOR X=1 TO 1
6:READ A:B$(X)=CHR$(A):NEXT X
440 POKE 623,1:POKE 54279,MEM-16:POKE
53277,3:POKE 704,4:POKE 705,198:POKE 7
07,51
450 FOR X=S TO S+108:POKE X,2:NEXT X:F
OR X=S+109 TO S+146:POKE X,1:NEXT X:FO
R X=S+147 TO S+255:POKE X,2:NEXT X
460 SOUND 1,160,8,2:GOTO 120
470 DATA 84,170,170,170,170,170,170,69
,84,255,255,255,255,255,255,69
480 DATA 14,14,14,4,31,31,14,14,14,14,
10,10,10,10,33,35,35,255,255,35,35,33,
160,128,128,255,255,128,128,160
```

# ROBOT DUNGEON

Co-Authors
Eric Anschuetz
Robert Anschuetz
John Weisgerber
[redacted]
Date: 6/12/84

For Atari Computers (all)
With a minimum of
24K cassette; 32K disk
Written in ATARI BASIC

ROBOT DUNGEON

ROBOT DUNGEON is known for its deadly robots, electrified walls, and teleports. Runor has it that no one has ever escaped alive! Nobody knows for sure, but the best estimations are that ROBOT DUNGEON is comprised of 3 levels of 400 rooms each, arranged in a 20 X 20 format.

The object of the game is to traverse from the lower-left room of each level to the upper-right room, all while amassing as many lives and as high a score as possible. Upon reaching the upper-right room of the 400 rooms on Level 1, you must touch the staircase and you will then procede automatically to walk down to the next most treacherous level, Level 2. At this point there is an animation sequence that will give you time to relax. Sit back and enjoy your figure as he jumps down the stairs, approaches the door, and slips through just in time!

Similar to what you did on Level 1, you must also reach the upper-right room of Level 2 (labeled room #800). Touch the staircase here and you will be rewarded with viewing the animation sequence once again, and will procede to the very dangerous Level 3!

If you manage to guide yourself to the upper-right room of Level 3 (labeled room #1200), and touch the escape key, you will be greeted with an ending animation sequence showing yourself leaving the dreaded ROBOT DUNGEON (and as you will see, apparently just in time!).

ROBOT DUNGEON is no ordinary dungeon, either! No, ROBOT DUNGEON is rather a 3 level <u>MAZE</u> with <u>only 1</u> escape route possible for each level. Therefore, it is <u>heartily recommended</u>

that you map each level. An example map is given in figure 1.

Throughout ROBOT DUNGEON there are various treasures and traps which with you must be acquainted with. The most abundent of these figures is the "SYMBOL OF LIFE". (It looks like a sideways figure eight). Touching this symbol will up your hit point total by one. The hit point total (which is foung on the lower-left of the score window) decreases by one every time you touch a robot, touch a wall, or touch a teleport (more on teleports later). It is very important to get the "SYMBOL OF LIFE" in each romm or you will have a very short adventure in ROBOT DUNGEON, indeed!

A symbol closely related to the "SYMBOL OF LIFE" is the "TREASURE CHEST". Within the "TREASURE CHEST" lies 5 hit points. Needless to say, don't pass up many of these!

Another sybol is the "ASSORTED GOLD". This pile is worth 250 points! Akin to the "ASSORTED GOLD" is the "DIAMOND", which is worth 1000 points!

Perhaps the most useful symbol is the deadly "ROBOT ZAPPER". This symbol looks like a zig-zag line and kills every robot in a room when touched!

Something to definitely stay away from are the dreaded "TELEPORTS". These are shaped like black pits and bring you back to the first room of the level which you are on! They also cost you one hit point! Stay away from these at all costs!

A summary of symbols is given in Figure 2 along with a sketch.

You are not defenseless as you wander ROBOT DUNGEON. You are equipped with a pistol which can be aimed in any direction. To shoot, point your joystick in a direction and press the fire

button. The killing of each blue robot is worth 10 points. Notice how I said blue robot. The orange ones are indestructible! On Level 1, there is one orange robot and 4 blue robots in each room. On the second level, there are 2 oranges and 3 blues. And on the third level, there are 3 oranges and only 2 blue ones!

You will note that as you pick up symbols, they are gone forever but robots are resurrected each time you enter or reenter a room.

By pressing any key during the course of play, you can freeze the action. Press another key and play resumes. You may also save the game in progress by pressing any one of the console keys (START, SELECT, or, OPTION). Note that on a disk system, saving a game will overwrite the previous version of your maze. If you want to have several different stages of your progress, you will have to change disks. Be sure to save your game very frequently!

When your hit point total reaches zero, your game is over. Pressing the start button will give you the choice of loading in a new dungeon or replaying the old one at the same point it was at when you began. If it was a saved game in progress, you start where the saved game started. However, all symbols that you had gotten since you began the session will now be gone.

To play a game of ROBOT DUNGEON, you first need to make a maze. To create a maze, you must use the MAZE CREATOR PROGRAM listed at the end of this article. This program will ask you to give a name to your dungeon. This is not the disk filename. Think of a name which you would like to call the dungeon, enter it, and press return. Next, you are asked to give a filename.

Cassette users simply enter "C:". Disk users should give "D:DUNJIN.DAT". After pressing return, there will be a long delay of approximately 15 minutes, as the maze is being created. Don't worry about the numbers being printed, but for those of you interested, the first column stands for the level, the second is the room number, and the third and fourth are used for determining a way out. After this is finished the program will prompt you to ready your disk drive or cassette recorder and you can now save your maze.

Now you can run the main program! This program will ask you for a filename. At this point, simply give the same one as you did in the creator. You then load the maze in by pressing return a couple times, and your game will start. Note that you do not have to create a new dungeon each time you wish to play the game. Simply use the same one over and over until you complete it (saving it periodically, of course). When you are done with one maze, you can then make another totally new one! Good luck!

Figure 1

MAP

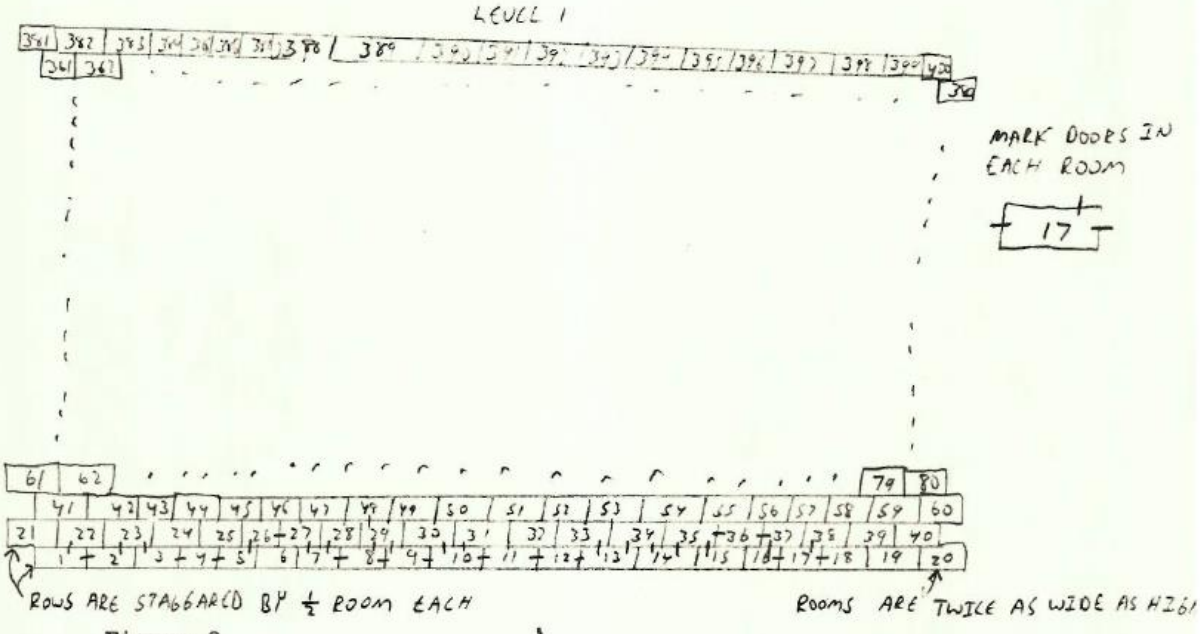
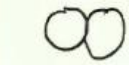


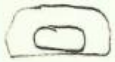

Figure 2

SYMBOLS

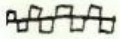

SYMBOL OF LIFE — 1 HIT POINT


ASSORTED GOLD — 250 POINTS


DIAMOND — 1000 POINTS


TELEPORT — BACK TO FIRST ROOM OF LEVEL


TREASURE CHEST — 5 HIT POINTS


ROBOT ZAPPER — KILLS EVERYTHING

# SPACE ASSAILANTS

Co-Authors
Eric Anschuetz
Robert Anschuetz
John Weisgerber
[redacted]
7/17/84

Atari BASIC
16K cassette
24K disk

Space Assailants

In the game Space Assailants, you are earth's lone defender in a battle which takes place in the midst of a treacherous meteor shower. You must protect the earth from an invading fleet of evil aliens, while avoiding the hurtling meteors. Your main concern is keeping the mothership, which leads the enemy armada, from landing. Although your weaponry cannot breach the powerful shields of the mothership, you can prevent its landing by destroying the accompanying fleet of alien vessels. The fleet drops each time it reaches a side of the playfield in its back and forth motion. After the fleet has dropped eight times, the mothership is considered to have landed, and your game is over. If you are able to destroy the fleet before the mothership lands, you will move on to the next round.

Rounds one through six consist of an alien fleet which increases in size from six to thirty-six at the rate of six per round. Rounds seven through twelve are exactly like one through six, but the aliens' speed is doubled. During rounds thirteen through eighteen, the aliens move slowly again, but the meteors travel at double speed. In rounds nineteen through twenty-four, both the aliens and the meteors move doubly fast. Sets after this remain at the same level of difficulty as sets nineteen through twenty-four.

If you are you are hit by a meteor, you lose one of three lives and must start your current round over. You score ten points for each alien vessel destroyed. Your score is located at the top and middle of the screen. To its left are lives remaining, and to its right is the round number.

You may only fire one missile at a time, and you must wait until the missile either hits an alien or clears the screen to fire again. As the rounds progress, and there are more aliens to contend with, you will find that the best strategy consists of destroying the end columns of assailants, thus lengthening the amount of time between drops of the fleet. Destroying the bottom row of aliens will not delay the mothership's landing.

Space Assailants combines several capabilities of the Atari into a successful, relatively fast action game. The least common of the capabilities used is antic's ability to switch quickly between two display lists. In the case of Space Assailants, this allows the meteors, which are in graphics four, and the aliens, which are in antic four, to be displayed at the same horizontal level, as if the meteors and aliens were present simultaneously. Since antic normally does sixty screen refreshes per second, each of the two display lists is used thirty times a second, or half as often as usual, so the quality of the screen image does suffer slightly, but this is negligible when cosidering the overall effect.

It is fairly simple to use two display lists instead of one on the Atari. The key is the jump on vertical blank, or JVB, insruction. This is the instruction placed at the end of a display list to tell antic that it can stop putting up new lines of graphics, and wait for vertical blank so it can jump up and begin its next screen refresh. The JVB instruction consists of three bytes. The first is a sixty-five, and simply tells antic that this is indeed a JVB. The last two are a vector memory address which points to the start of the display list

where antic will jump when it begins the next refresh.

To make antic use two display lists, the first step is to write two display lists into memory, and have the JVB of the first point to the second, and the JVB of the second point to the first. At first glance, it seems this alone would cause antic to switch between two display lists. However, at least while using Atari BASIC, antic doesn't use the address given as part of the JVB. Instead, as soon as it sees the sixty-five of the JVB, antic refers to locations 560 and 561. This creates the problem of getting antic to forget what's in 560 and 561, and look at the JVB addresses. Fortunately, there's a way to do this very easily from BASIC. Simply poking location 54286, the location normally used to enable display list interrupts, to zero, causes the Atari to stop referring to several registers in the first K of memory, including 560 and 561. After antic stops using 560 and 561 to find the start of the display list, the only place it can go for the information is the address given in the JVB.

Poking 54286 to zero solves the problem of having antic switch between our display lists, but it also creates several new problems due to some of the other registers which cease to function normally. For example, after poking 54286 to zero, location 756 can no longer be used to switch character sets, and the Atari virtually forgets the meaning of the BASIC functions stick and strig.

There are, as you can see by playing Space Assailants, ways to get around these problems. To get a response from the stick or strig functions, you must simply refer to the

hardware registers. For example, STRIG(0) is equivalent to PEEK(53264), and STICK(0) is similar to PEEK(54016). In the case of the STICK function, there is a slight difference between what the value of the function would be and the value in the hardware register, but it's easy enough to figure out the proper values for the various stick positions: simply add 240 to whatever the STICK value for that position would be. For instance, if the stick is centered, the value of STICK(0) is fifteen, but PEEK(54016) is equal to 255.

In order to switch between two character sets while 54286 is poked to zero, the equivalent of the hardware register for 756 is used. This register is located at 54297 and is normally set to 255. Actually, a split second after you poke this location to some other value, it is reset at its original value and, while the character set will flash to the new value for a sixtieth of a second (one screen refresh), as soon as the computer checks the value in 756 it will look for the character set in the position indicated there. This is why you must normally use 756 rather than 54297 to change character sets. After 54286 is set to zero, the computer never refers to 756, so the computer will continue to look for the character set in whatever position you may have poked, if only for a moment, in 54297. Just don't try to find out where you have placed the character set by peeking 54297, because it will already have been reset to 255. The case for register 559, which is the control register for antic, is similar to that for 756.

Another problem created by poking 54286 to zero is

that setcolors and the registers for the players colors are not reffered to. In Space Assailants, the desired colors were simply set before poking 54286 to zero, but there are surely hardware registers which can be used to change colors around while 54286 is set at zero.

That just about covers the problem of using two display lists in a game written purely in Atari BASIC, but there are several other portions of the Space Assailants program which will interest you. First, there is the movement of the meteors and of the fleet of aliens. This movement is achieved, primarily, by changing the value of the load memory scan or LMS instruction in each of the display lists. In the case of the meteors, since there are ten bytes in a line of graphics four, the value in the LMS is incremented by ten or twenty per loop, depending on the skill level. In the case of the fleet, the LMS is used only to create the horizontal movement and is incremented by either .25 or .5 per loop, which means at most the fleet moves in every other loop. As an indicator for when the fleet hits the side and must reverse direction and drop, two players have been placed at either side of the playfield. Whenever the value in the cllision register for either of these players is greater than one (a star placed randomly behind one of the players may cause a value of one), the value of a variable which was being added to the value in the fleet's LMS is made the negative of what it previously was. This causes the fleet to reverse direction. Simultaneously, one of eight blank line instructions in the fleet's display list is converted from zero to 112, thus

moving the assailants down on the screen. The next time the fleet hits a side the next blank line instruction will be converted until, after eight drops, the game ends, or until the set is cleared, at which time all the blank line instructions are reset at zero for the next round.

Another interesting part of the program deals with the movement of the shot up the screen. While vertical movement of players is normally implemented from BASIC by the use of machine language subroutines, in Space Assailants it is accomplished entirely from BASIC itself. The method here is to interpret the player's data as a string of characters, and to print the player into the desired spot in memory. The registers of 88 and 89 comprise a vector memory location for the place where the PRINT function begins, or what the computer interprets as POSITION 0,0. By poking 89 to the page number at which the desired player is located, you allow yourself to control the vertical position of the player (which is controlled by its position on the page, 0-255) by poking 88 to the desired location and printing the player there. In order to create vertical movement as in the case of the shot, you simply increment the value which will be poked into 88, position the computer to print at 0,0 to make sure you will print at the exact address you desire and print the player. In the case of the shot in Space Assailants, the character string is "88\D ". The reason for the blanks is that the shot is incremented by seven, so each time the shot is moved up and reprinted, it automatically erases the previous shot by printing blanks where it had been located. The same

method could be used for downward movement if the string were preceded by blanks. Also, this method could be used to make a player shift through several positions by using several different strings and printing them into the same point in memory.

Along the same general lines as the shot movement is the method used in Space Assailants to create the two players placed along either side to detect collisions with the fleet as previously noted. To begin with, all 256 bytes of the pages set aside for these players are poked to 255. The string A$ or "SPACE ASSAILANTS" is then used to put the title of the game onto both players. Rather than trying to understand the method for doing this from a written description, you're better off referring to lines 550 and 560 of the program. You should be able to follow this method with the knowledge that 32 is subtracted from the peek of ADR(A$(X)) in order to give the CHR$ value, and location 57344 is the start of character data in the Atari's memory.

The style of programming used in Space Assailants is slightly different than that used in many other programs in that exact addresses are used for the player/missile base, the redefined character sets, and several other items placed in the four K of memory used for such things. This is sometimes much easier than setting back a certain number of pages based on the peek of location 106 and usually uses somewhat less memory. If you are writing a program which will fit into 16K anyway, why not just set all the addresses as if it will always be run in 16K? The program will be just as universal as if you rely on PEEK(106), and exact addresses

shorten the code others will have to type in.

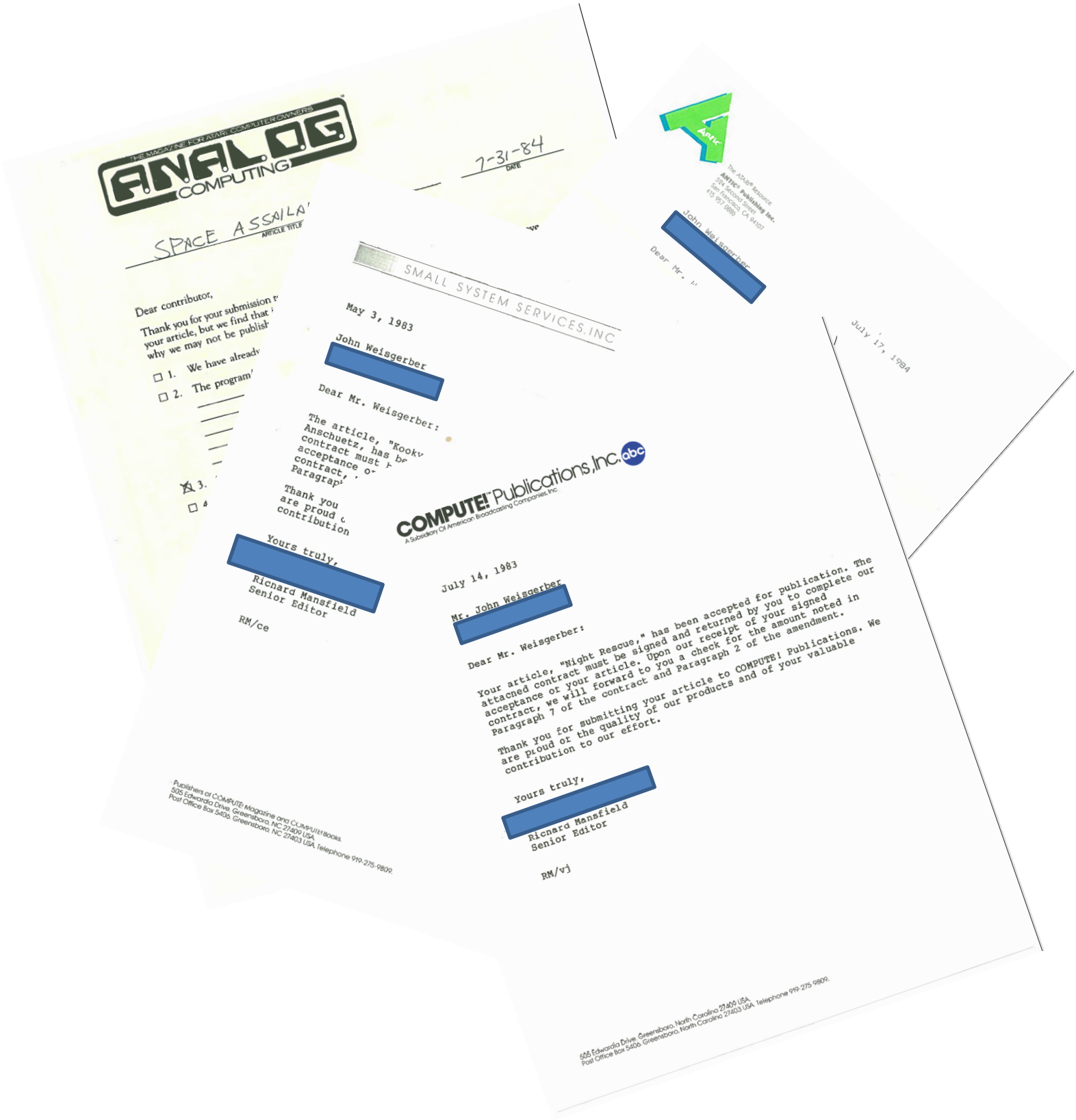
If you don't have the time or patience to type in Space Assailants, just send a self-addressed, stamped mailer, a blank cassette, and $3.00 to John Weisgerber, [redacted] Be sure to include the name of the program you want.

```
100 GOTO 420
110 SOUND 0,0,0,0:NSA=0:SE=SE+1:POKE 656,0:POKE 657,34:? SE;"  ":IF LI=0 THE
N 280
120 FOR Y=13262 TO 13271:POKE Y,0:NEXT Y:POKE 54286,0:POKE 13273,50:LM=50:PO
KE 13163,240:X=240:DR=13262:F=0
130 FOR Z=12859 TO 13099 STEP 4:POKE Z,0:POKE Z+1,0:NEXT Z
140 R=INT((SE-1)/6):NL=SE-R*6:IF SE>6 THEN Q=0.5:IF SE>12 THEN SD=20:Q=0.25
150 RESTORE 720:FOR L1=13059 TO 13095-NL*40 STEP -40:READ TA:FOR L2=L1 TO L1
+20 STEP 4:POKE L2,TA:POKE L2+1,TA+1
160 NSA=NSA+1:NEXT L2:NEXT L1:Q=ABS(Q):POKE 53278,0:IF SE>18 THEN Q=0.5
170 X=X-SD:POKE 13163,X:IF X=0 THEN X=240
180 IF F=1 THEN R=R-7:GOSUB 300
190 IF X/80=INT(X/80) THEN POKE 54297,CH:CH=48*(CH=60)+60*(CH=48)
200 IF PEEK(54016)=251 THEN YP=YP-(YP>53)*4:POKE PLX,YP
210 IF PEEK(54016)=247 THEN YP=YP+(YP<185)*4:POKE PLX,YP
220 IF PEEK(53264)=0 AND F=0 THEN F=1:R=204+(DR-13262):POKE PLX+1,YP+4:SP=YP
+4
230 IF F=1 THEN R=R-7:GOSUB 300
240 IF PEEK(53254)>1 OR PEEK(53255)>1 THEN Q=-Q:POKE DR,112:DR=DR+1:IF DR=13
272 THEN 280
250 LM=LM+Q:POKE 13273,LM
260 IF PEEK(53252)>0 THEN 400
270 POKE 53278,0:GOTO 170
280 SETCOLOR 2,0,0:SETCOLOR 0,8,4:POKE 54286,255:POKE 560,TR01:POKE 561,TR02
:FOR X=PLX TO PLX+3:POKE X,0:NEXT X
290 POKE 53278,0:POKE 84,0:? #6;"     ":SOUND 2,0,0,0:SOUND 3,0,0,0:GOTO 600
300 IF PEEK(53253)>1 AND PEEK(53253)<8 THEN JW=7:GOTO 340
310 POKE 88,R:POKE 84,0:? #6;"88\D        ":IF R<50 THEN POKE 84,0:? #6;"
";F=0
320 IF PEEK(53253)>1 AND PEEK(53253)<8 THEN JW=0:GOTO 340
330 RETURN
340 HP=12818+INT((((R-40)+JW-(DR-13262)*7)/14)+0.5)*40+INT(((SP-48)/16+(LM-5
8)/4)*4):IF HP>13090 THEN RETURN
350 EA=0:FOR Z=HP TO HP+3:EA=EA+PEEK(Z):POKE Z,0:NEXT Z:IF EA=0 THEN R=R+JW:
RETURN
360 POKE 84,0:? #6;"     "
370 SOUND 0,220,10,15
```

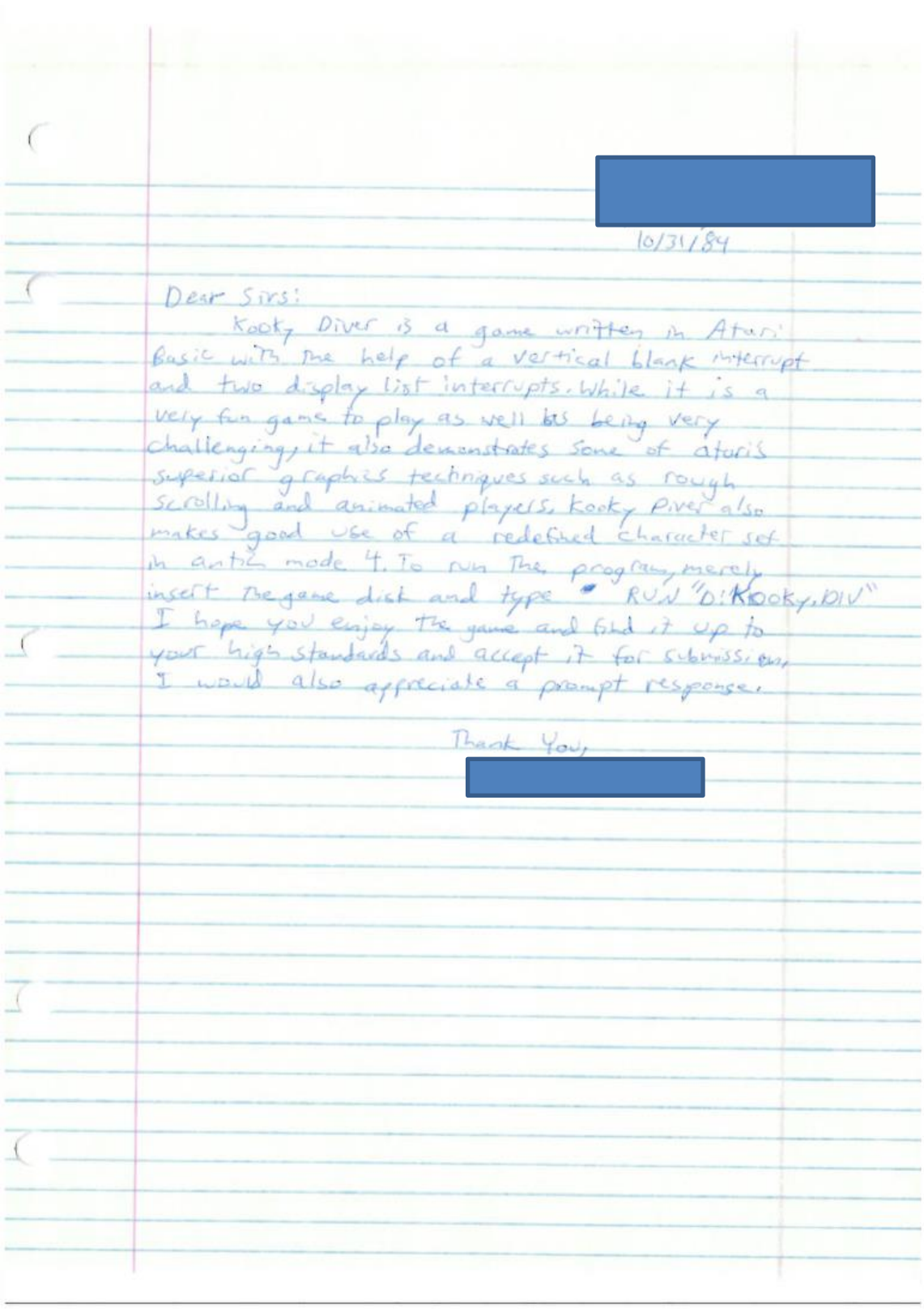
```
380 F=0:NSA=NSA-1:SC=SC+10:POKE 656,0:POKE 657,18:? SC;:IF NSA=0 THEN POP :G
OTO 110
390 SOUND 0,0,0,0:RETURN
400 LI=LI-1:POKE 656,0:POKE 657,5:? LI;:POKE 84,0:? #6;"    ":SE=SE-1:FOR Z=
200 TO 255:SOUND 0,Z,8,15:NEXT Z
410 GOTO 110
420 GRAPHICS 2:TRO1=PEEK(560):TRO2=PEEK(561):TRODL=TRO1+256*TRO2:SETCOLOR 2,
0,0:SETCOLOR 0,8,4
430 POKE TRODL+10,6:POKE TRODL+11,6:POKE TRODL+12,6:POKE TRODL+13,2:POKE 16,
64:POKE 53774,64
440 POSITION 2,4:? #6;"SPACE ASSAILANTS":POSITION 5,8:? #6;"ANSCHUETZ/WEISGE
RBER/ANSCHUETZ"
450 POSITION 4,6:? #6;"initializing"
460 CH1=48*256:CH2=60*256:FOR I=0 TO 511:POKE CH1+I,PEEK(57344+I):POKE CH2+I
,PEEK(57344+I):NEXT I
470 FOR I=8 TO 55:READ A:POKE CH1+I,A:NEXT I:FOR I=8 TO 55:READ A:POKE CH2+I
,A:NEXT I
480 FOR I=56 TO 79:READ A:POKE CH1+I,A:POKE CH2+I,A:NEXT I:POKE 756,60:CH=60
490 FOR MIAD=56*256 TO 58*256:POKE MIAD,0:NEXT MIAD
500 FOR M1=1 TO 10:M2=INT(RND(0)*220)+14336:R=INT(RND(0)*7)+1:POKE M2,2^R:PO
KE M2+240,2^R:NEXT M1
510 FOR I=13256 TO 13291:READ A:POKE I,A:NEXT I:FOR I=13156 TO 13165:READ A:
POKE I,A:NEXT I
520 FOR I=13166 TO 13210 STEP 2:POKE I,9:POKE I+1,48:NEXT I:FOR I=13211 TO 1
3213:READ A:POKE I,A:NEXT I:DIM A$(20)
530 POKE 13260,PEEK(660):POKE 13261,PEEK(661):POKE 13160,PEEK(660):POKE 1316
1,PEEK(661):A$="SPACE ASSAILANTS"
540 PM=48*256:PLX=53248:FOR X=216 TO 223:READ A:POKE PM+1024+X,A:NEXT X:FOR
X=PM+1536 TO PM+2047:POKE X,255:NEXT X
550 FOR X=1 TO 16:A=PEEK(ADR(A$(X)))-32:FOR Y=0 TO 7:POKE X*8+Y+PM+1576,255-
PEEK(A*8+Y+57344)
560 POKE X*8+Y+PM+1832,255-PEEK(A*8+Y+57344):NEXT Y:NEXT X:POKE 704,50:POKE
705,40:POKE 706,130:POKE 707,130
570 POKE 559,62:POKE 623,1:POKE 53277,3:POKE 54279,PM/256:POKE 53256,1
580 POKE 15880,199:POKE 15881,200:POKE 15882,200:POKE 15883,201
590 POSITION 0,6:? #6;"press start to begin":POKE 89,53:POKE 82,0:POKE 755,0
600 FOR RA=12859 TO 13099 STEP 4:POKE RA,0:POKE RA+1,0:NEXT RA
```

```
610 IF PEEK(53279)<>6 THEN 610
620 POKE 560,100:POKE 561,51:SETCOLOR 0,12,8:SETCOLOR 1,3,6:SETCOLOR 2,8,2:S
ETCOLOR 3,12,4
630 POKE PLX+2,48:POKE PLX+3,200:POKE PLX,121:YP=121:Q=0.25:SE=0:SC=0:NSA=0:
SD=10:LI=3
640 POKE 656,0:POKE 657,5:? LI;:POKE 657,18:? SC;"      ":SOUND 2,190,10,3:S
OUND 3,192,10,3:GOTO 110
650 DATA 3,15,59,63,15,4,16,65,192,240,236,252,240,16,64,0,0,67,79,30,14,14,
15,20
660 DATA 0,193,241,180,176,176,240,20,0,4,19,63,62,30,79,16,0,16,196,252,188
,180,241,4
670 DATA 3,15,62,63,15,4,1,0,192,240,188,252,240,16,4,65,0,3,15,15,30,79,79,
5
680 DATA 0,192,240,240,180,241,241,80,16,4,3,63,62,30,31,20,4,16,192,252,188
,180,244,20
690 DATA 0,0,0,0,11,63,127,227,0,0,60,255,153,153,255,24,0,0,0,0,208,252,254
,199,112,112,112,66,0,0,0,0,0,0,0,0
700 DATA 0,0,0,0,68,50,50,96,4,96,4,96,4,96,4,96,4,112,70,0,62,65,100,51,112
,112,112,66,0,0,73,0,56,48,65,200,51
710 DATA 8,8,28,28,62,62,42,42
720 DATA 1,3,5,1,3,5
```

# A/W/A MAGAZINE CORRESPONDENCE



# KOOKY DIVER



[redacted]

10/31/84

Dear Sirs:

Kooky Diver is a game written in Atari Basic with the help of a vertical blank interrupt and two display list interrupts. While it is a very fun game to play as well as being very challenging, it also demonstrates some of ataris superior graphics techniques such as rough scrolling and animated players. Kooky Diver also makes good use of a redefined character set in antic mode 4. To run the program, merely insert the game disk and type " RUN "D:Kooky.DIV" I hope you enjoy the game and find it up to your high standards and accept it for submission. I would also appreciate a prompt response.

Thank You,

[redacted]

In the game KOOKY DIVER, you play the role of a diver who must search the sea for sunken treasure and escape from the depths alive! On your dives, you will encounter black pearls, treasure chests, pistols, vases, steering wheels, and many other treasures. But what Kooky doesn't realize as he begins his expedition is that not only are there a dangerous shark and a school of man-eating piranha to block his path, but that a giant clam has swallowed all of the treasure!

You, as Kooky, will have to dive for the various treasures and attempt to bring them up to the salvage boat. To move Kooky in any particular direction, just move the joystick in the corresponding direction. To tread water, push the joystick in any of the diagonal directions.

You must time it just right so that the clam's mouth is open when you touch the treasure in order to have a chance to make a successful salvage. If the mouth is open and you are touching the treasure, merely press the fire button on the joystick and you will have picked up the treasure. Of course, if you touch any part of the clam when its mouth is closed, you will lose a life. After you have a treasure in your possession, you will use up your limited supply of oxygen rather quickly.

Luckily, there are three coral reefs for you to deposite your treasure on while surfacing for more oxygen. Provided that your current oxygen level is below 50 units, you will completely fill your tanks every time you bring your head out of the water. To drop off a treasure on any reef, all you must do is make sure Kooky is only touching the yellow portion of the reef. At this point, the treasure will automatically be set there with no further actions required. To pick up a treasure again, simply press the fire button while touching the treasure as before.

Of course, any collision with the piranha or the shark will cause instant death and send Kooky plummetting toward the ocean floor. After three deaths, your game will end and the high score will be shown. At this point, you can press the "START" button on the computer to start a new game with three new divers. You may end up doing this quite a few times in the beginning because after all, nobody ever said that diving was easy!

The ultimate goal of the game is to deposite your treasure in the boat that is moving across the top of the screen. To do this, just touch the boat and your treasure will appear inside. At this point, a new treasure will

appear inside the clam, and you can now try to get this treasure to the boat. Have fun diving!


WORCESTER, MA 016
WORCESTER, MA 016
U.S. POSTAGE
THE MAGAZINE FOR ATARI COMPUTER OWNERS
ANALOG
COMPUTING
565 MAIN STREET CHERRY VALLEY, MA 01611
ERIC ANSCHUETZ

THE MAGAZINE FOR ATARI COMPUTER OWNERS

ANALOG COMPUTING

KOOKY DIVER — ARTICLE TITLE

11-19-84 — DATE

Dear contributor,

Thank you for your submission to **ANALOG Computing**. Our technical staff and editors have evaluated your article, but we find that it does not meet our needs at the present time. There are several reasons why we may not be publishing your article:

☐ 1. We have already purchased a similar article, or have one under development.

☐ 2. The program/documentation needs improvement:
VARIABLE PX IN 720 NOT USED.

☒ 3. The article has limited appeal to our readers.

☐ 4. The article needs re-writing:

☐ 5. The article is not appropriate to our needs.

Thank you for submitting your article/ideas to ANALOG Computing.

[redacted]
AUTHORIZED SIGNATURE
Lee H. Pappas
Publisher, ANALOG Computing

[redacted]
AUTHORIZED SIGNATURE
Charles Bachand
Technical/Software, ANALOG Computing

Additional comments:
A LITTLE TOO "BEGINNER LEVEL" FOR US.
SORRY...

# KOOKY KLIMBER

Robert and Eric Anschuetz
[redacted]

John Weisgerber
[redacted]

2/23/83

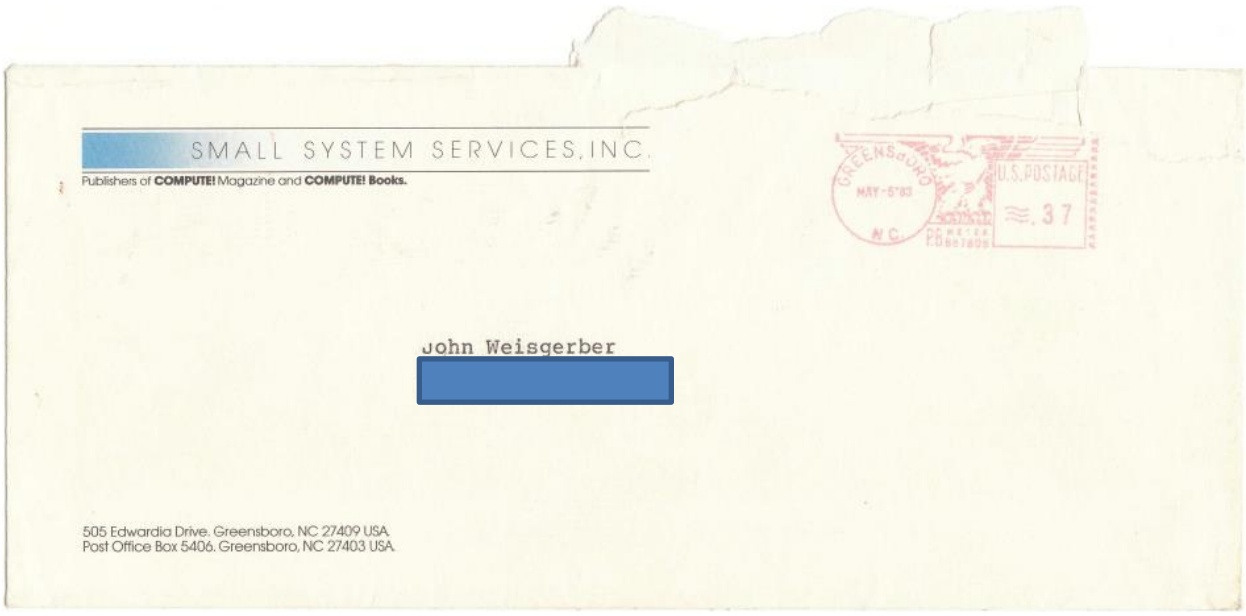
Mr. Robert C. Lock
Publisher/Editor-In-Chief
COMPUTE! Magazine
Post Office Box 5406
Greensboro, Nc 27403

Dear Mr. Lock:

We, the undersigned, are three 17 year old high school seniors. Our first experience with COMPUTE! occurred in November, 1981. We had walked into a local Computerland store and we noticed on the monitor screen a very attractive looking game. We asked the person occupying the seat in front of the computer about the program. He stated that its title was "SHOOT" and that he had programmed it from an article in a magazine called COMPUTE!. We quickly purchased the magazine with the knowledge that we would recieve an ATARI 400 computer for Christmas that year and that we would need exciting games for it.

Well, sir, the point of this letter is that a well programmed, attractive looking program influenced us into purchasing our first COMPUTE! and we have not missed a single copy since. Undoubtedly, many others must have started their COMPUTE! experience with the anticipation of playing a fast arcade style game. We three computer fanatics are convinced that our program, "KOOKY KLIMBER", can accomplish the same with other readers.

We have been programming ever since that Christmas Day in 1981 and have been gaining experience continually. One of the major helps for us in our advancement on the programming ladder is COMPUTE! magazine. In fact, "KOOKY KLIMBER" would not have been possible were it not for a previous article that appeared in COMPUTE! describing vertical player movement and player animation. For this very reason, we feel that you should seriously consider publishing our article and program. Programmers such as ourselves, and even beginning programmers, can increase their knowledge of some of the special features exclusive to the ATARI such as an understandable approach to rough scrolling and a use of easily moveable players. We have written the enclosed article in such a manner that programmers with little to no prior knowledge can learn these techniques and so that advanced programmers, as well, can benifit.


SMALL SYSTEM SERVICES, INC.
Publishers of COMPUTE! Magazine and COMPUTE! Books.
GREENSBORO
MAY-5'83
N.C.
U.S.POSTAGE
.37
John Weisgerber
[redacted]
505 Edwardia Drive. Greensboro, NC 27409 USA.
Post Office Box 5406. Greensboro, NC 27403 USA.

SMALL SYSTEM SERVICES, INC

May 3, 1983

John Weisgerber
[redacted]

Dear Mr. Weisgerber:

The article, "Kooky Klimber," by you and Robert and Eric Anschuetz, has been accepted for publication. The attached contract must be signed and returned by you to complete our acceptance or your article. Upon our receipt of your signed contract, we will forward to you a check for the amount noted in Paragraph 7 of the contract and Paragraph 2 of the amendment.

Thank you for submitting your article to COMPUTE! Publications. We are proud of the quality of our products and of your valuable contribution to our effort.

Yours truly,

[redacted]

Richard Mansfield
Senior Editor

RM/ce

Publishers of COMPUTE! Magazine and COMPUTE! Books.
505 Edwardia Drive, Greensboro, NC 27409 USA.
Post Office Box 5406, Greensboro, NC 27403 USA. Telephone 919-275-9809.

This contract, when accepted by you, constitutes an agreement between Small System Services, Inc. (hereafter **COMPUTE!**) and you, with respect to the work tentatively entitled

May 3, 1983

Kooky Klimber

(hereinafter "the Work"), as follows:

1. You hereby grant to **COMPUTE!**, its subsidiaries, and licensees exclusive worldwide periodical rights in and to the Work including the right to print the Work in a book or books exerciseable by notice to you and accompanied by payment of an amount equal to 15% of the amount you were first paid for the use of your material.

2A. In addition to this amount, **COMPUTE!** warrants that use of your Work in a book or collected form will produce quarterly royalty payments to you based on the net sales of the book by **COMPUTE!**. These payments will be calculated on the basis of your number of pages and fractional pages in the book as a percentage of the total number of pages in the book. A royalty in an amount corresponding to 7.5% of the net sales of the book will be allocated pro rata among all of the authors contributing to the book.

B. It is understood as a part of this agreement that **COMPUTE!** has the full right to disseminate software contained in the Work in electronic form as a part of the normal distribution of the magazine and its contents. In the event such dissemination generates additional net sales in electronic form, you shall be paid additional royalties as defined in Paragraph 2A above.

3. You also grant **COMPUTE!** the right to condense and adapt the Work for publication in all languages throughout the world and to license all of the rights granted hereunder to third parties. You also grant the right to reprint the Work in promotional materials and reprints.

All other rights in and to the Work remain your property, except that you agree that **COMPUTE!** has the right of first publication in any form in any country in the world.

4. **COMPUTE!** shall be the copyright owner of the exclusive rights purchased herein.

5. **COMPUTE!** shall have the right to use the name, biography, and likeness of the author in connection with the publication and promotion of the Work.

6. You warrant that you own the rights conveyed herein or, as the owner's duly authorized Agent, that you have the full power to grant such rights, that the Work does not infringe any copyright or rights of any third party, or contain any matter that is libelous or otherwise in contravention of law.

7. In full and complete consideration for the rights herein granted, **COMPUTE!** will pay you the sum of: $200.00

This sum does not preclude, and is not affected by, the additional amounts provided for in Clauses 1 and 2 above, should republication of the work occur.

If the foregoing meets with your approval, please acknowledge your agreement by signing and returning one copy of this contract at your earliest convenience.

Agreed and approved:

Very truly yours,

Author or authorized agent John Weisgerber Robert Anschuetz Eric Anschuetz

Social Security Number

Kathleen Martinek, Managing Editor
by authority of the President

This document constitutes an amendment to the foregoing contract, and is a part of it. Because of the exceptional quality of the software contained in your Work, as previously defined, **COMPUTE!** wishes to purchase the software rights to your Work as well. In consideration of these additional rights granted by you, we will pay you the following residual amounts:

1. Upon publication of your software in electronic media form, **COMPUTE!** will pay you the sum of 15% of net sales generated by your software. If your software is published electronically as a portion of a collection of such software, **COMPUTE!** will pay you based on the percentage (by size) that your Work comprises of that collection.

2. In consideration of our additional payment to you, upon your acceptance of this contract, of $200 and future royalties, you hereby agree to grant us these exclusive rights in the sale, marketing, and distribution of your software.

3. Royalties, after publication in electronic form, will be paid to you on a quarterly basis.

4. Should **COMPUTE!** fail to produce, and market your software in electronic form within 1 (one) year from the date of publication, you may withdraw your acceptance of this amendment, and all rights herein granted shall revert to you. Such notice from you, which must be in writing at any time after expiration of the one-year period, is without any other constraint, and the initial payment of $200 for these rights is understood to be non-refundable.

As amended: _____

Kathleen Martinek, Managing Editor, 5/3/89
by authority of the President

Agreed and approved:

_____

Author or authorized agent

Date: _____

COMPUTE!™ Publications, Inc. abc
One of the ABC Publishing Companies

September 6, 1985

Mr. John Weisgerber
[redacted]

Dear Mr. Weisgerber:

Following a recent inventory of our article pool, we have concluded that some of the articles therein are unlikely to become published within a reasonable amount of time.

Your article, "Kooky Klimber" is included. There are several reasons why an article becomes unlikely to be published: overlap with other published articles; the material is no longer timely, etc.

This notification constitutes a release of your material and you are free to submit it elsewhere, and may, of course, keep the fees we paid you for it. Because it was originally accepted here, we consider your work to be of exceptional quality and would encourage you to submit articles to us in the future.

Yours truly,

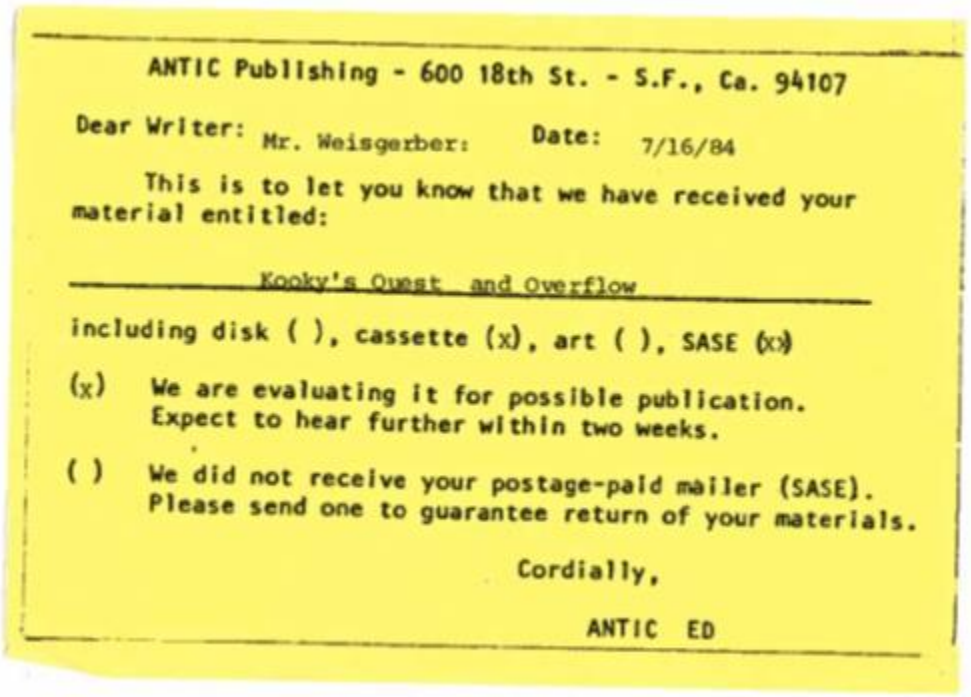
[redacted]

Richard Mansfield
Senior Editor

RM/jk

324 West Wendover Avenue, Suite 200. Greensboro, North Carolina 27408 USA.
Post Office Box 5406. Greensboro, North Carolina 27403 USA. Telephone 919-275-9809.
Telex Number 574411 WUPUBTLX GBO

# KOOKY'S QUEST



ANTIC Publishing - 600 18th St. - S.F., Ca. 94107

Dear Writer: Mr. Weisgerber: Date: 7/16/84

This is to let you know that we have received your material entitled:

Kooky's Quest and Overflow

including disk ( ), cassette (x), art ( ), SASE (x)

(x) We are evaluating it for possible publication. Expect to hear further within two weeks.

( ) We did not receive your postage-paid mailer (SASE). Please send one to guarantee return of your materials.

Cordially,

ANTIC ED

COMPUTE!™ Publications, Inc. abc
One of the ABC Publishing Companies

Re: Kooky's Quest

Dear Author:

We, at COMPUTE!, regret the delay in responding to your submission. However, after reviewing it we feel that it does not meet our present needs.

We are always interested, though, in reviewing programs and articles for possible publication. Please find enclosed a copy of COMPUTE!'s Author Guide which will assist you in submitting your material. Please do not hesitate to send us something in the future.

Thank you for thinking of COMPUTE!.

Yours truly,



Dan Carmichael
Submissions Editor

DC/vj

324 West Wendover Avenue, Suite 200. Greensboro, North Carolina 27408 USA.
Post Office Box 5406. Greensboro, North Carolina 27403 USA. Telephone 919-275-9809.

The ATARI® Resource
**ANTIC® Publishing Inc.**
524 Second Street
San Francisco, CA 94107
415 957 0886

John Weisgerber
[redacted]

Date: 7/17/84

Thank you for your recently submitted material entitled:

Kooky's Quest including

☒ listings, ☐ disk, ☒ cassette, ☐ art, ☒ SASE

☒ Your material has been accepted for publication in our

~~Game~~ department, or as a _____ now planned

for the _____ issue, or for other use within one year. (X)

Payment will be made upon publication. If we do not use this within one year, we will pay a "kill fee" and copyright to the material reverts to you. If we need a biographical statement or copyright agreement from you, a form is enclosed. Please return these to us as soon as possible.

---

☐ Sorry, we can't accept your material. Unfortunately, we don't have room to print all the good submissions we receive.

☐ Your material is being returned to you herewith, or
☐ is being held here pending your postage-paid mailer.

The volume of our editorial work prevents a more personal response. We appreciate your efforts and are glad that you submitted to Antic. Please keep us in mind in the future.

Cordially,
[redacted]
Nat Friedland,
Editor

( )Enc.

# NIGHT RESCUE

John Weisgerber
[redacted]

Richard Mansfield
Senior Editor
COMPUTE! Magazine

Dear Mr. Mansfield:

Eric and Robert Anschuetz and I have written a second game entitled "Night Rescue" which we feel is much more impressive than our previous effort, "Kooky Klimber". This game utilizes horizontal scrolling along with player/missile graphics and Atari's four color character modes, antic four and five. Because this game uses so many features unique to the Atari computer we feel it would serve as an exellent program to show others the various techniques useful in implementing these features from the BASIC programming language. The program itself is fairly well written, and the game is actually fun to play, something not true of many magazine games we have typed in recently. For these reasons we urge you to promptly accept this new offering for publication.

For our last article and game we were payed a total of $400. Since "Night Rescue" is far superior in all aspects to "Kooky Klimber", we feel a fair payment would be in the area of $750 ($250 each, if you prefer). We have already started our next game and have ideas for several others, so you will be hearing from us again. In the meantime, if you need to contact us, don't hesitate to call one of the Anschuetz at the number on the title page of the enclosed article.

Sincerely,

[redacted]

COMPUTE!™ Publications, Inc. abc
A Subsidiary Of American Broadcasting Companies, Inc.

July 14, 1983

Mr. John Weisgerber
[redacted]

Dear Mr. Weisgerber:

Your article, "Night Rescue," has been accepted for publication. The attached contract must be signed and returned by you to complete our acceptance of your article. Upon our receipt of your signed contract, we will forward to you a check for the amount noted in Paragraph 7 of the contract and Paragraph 2 of the amendment.

Thank you for submitting your article to COMPUTE! Publications. We are proud of the quality of our products and of your valuable contribution to our effort.

Yours truly,

[redacted]

Richard Mansfield
Senior Editor

RM/vj

505 Edwardia Drive, Greensboro, North Carolina 27409 USA.
Post Office Box 5406, Greensboro, North Carolina 27403 USA. Telephone 919-275-9809.

This contract, when accepted by you, constitutes an agreement between Small System Services, Inc. (hereafter **COMPUTE!**) and you, with respect to the work tentatively entitled

July 14, 1983

Night Rescue

(hereinafter "the Work"), as follows:

1. You hereby grant to **COMPUTE!**, its subsidiaries, and licensees exclusive worldwide periodical rights in and to the Work including the right to print the Work in a book or books exerciseable by notice to you and accompanied by payment of an amount equal to 15% of the amount you were first paid for the use of your material.

2A. In addition to this amount, **COMPUTE!** warrants that use of your Work in a book or collected form will produce quarterly royalty payments to you based on the net sales of the book by **COMPUTE!**. These payments will be calculated on the basis of your number of pages and fractional pages in the book as a percentage of the total number of pages in the book. A royalty in an amount corresponding to 7.5% of the net sales of the book will be allocated pro rata among all of the authors contributing to the book.

B. It is understood as a part of this agreement that **COMPUTE!** has the full right to disseminate software contained in the Work in electronic form as a part of the normal distribution of the magazine and its contents. In the event such dissemination generates additional net sales in electronic form, you shall be paid additional royalties as defined in Paragraph 2A above.

3. You also grant **COMPUTE!** the right to condense and adapt the Work for publication in all languages throughout the world and to license all of the rights granted hereunder to third parties. You also grant the right to reprint the Work in promotional materials and reprints.

All other rights in and to the Work remain your property, except that you agree that **COMPUTE!** has the right of first publication in any form in any country in the world.

4. **COMPUTE!** shall be the copyright owner of the exclusive rights purchased herein.

5. **COMPUTE!** shall have the right to use the name, biography, and likeness of the author in connection with the publication and promotion of the Work.

6. You warrant that you own the rights conveyed herein or, as the owner's duly authorized Agent, that you have the full power to grant such rights, that the Work does not infringe any copyright or rights of any third party, or contain any matter that is libelous or otherwise in contravention of law.

7. In full and complete consideration for the rights herein granted, **COMPUTE!** will pay you the sum of: $200.00

This sum does not preclude, and is not affected by, the additional amounts provided for in Clauses 1 and 2 above, should republication of the work occur.

If the foregoing meets with your approval, please acknowledge your agreement by signing and returning one copy of this contract at your earliest convenience.

Agreed and approved:

Author or authorized agent John Weisgerber

Social Security Number

Very truly yours,

Kathleen Martinek, Managing Editor
by authority of the President

This document constitutes an amendment to the foregoing contract, and is a part of it. Because of the exceptional quality of the software contained in your Work, as previously defined, **COMPUTE!** wishes to purchase the software rights to your Work as well. In consideration of these additional rights granted by you, we will pay you the following residual amounts:

1. Upon publication of your software in electronic media form, **COMPUTE!** will pay you the sum of 15% of net sales generated by your software. If your software is published electronically as a portion of a collection of such software, **COMPUTE!** will pay you based on the percentage (by size) that your Work comprises of that collection.

2. In consideration of our additional payment to you, upon your acceptance of this contract, of $200 and future royalties, you hereby agree to grant us these exclusive rights in the sale, marketing, and distribution of your software.

3. Royalties, after publication in electronic form, will be paid to you on a quarterly basis.

4. Should **COMPUTE!** fail to produce, and market your software in electronic form within 1 (one) year from the date of publication, you may withdraw your acceptance of this amendment, and all rights herein granted shall revert to you. Such notice from you, which must be in writing at any time after expiration of the one-year period, is without any other constraint, and the initial payment of $200 for these rights is understood to be non-refundable.

As amended: _____
Kathleen Martinek, Managing Editor, 7/14/83
by authority of the President

Agreed and approved: _____
Author or authorized agent

Date: 7-26-83

COMPUTE!™ Publications, Inc. abc
One of the ABC Publishing Companies

September 6, 1985

Mr. John Weisgerber
[redacted]

Dear Mr. Weisgerber:

Following a recent inventory of our article pool, we have concluded that some of the articles therein are unlikely to become published within a reasonable amount of time.

Your article, "Night Rescue" is included. There are several reasons why an article becomes unlikely to be published: overlap with other published articles; the material is no longer timely, etc.

This notification constitutes a release of your material and you are free to submit it elsewhere, and may, of course, keep the fees we paid you for it. Because it was originally accepted here, we consider your work to be of exceptional quality and would encourage you to submit articles to us in the future.

Yours truly,

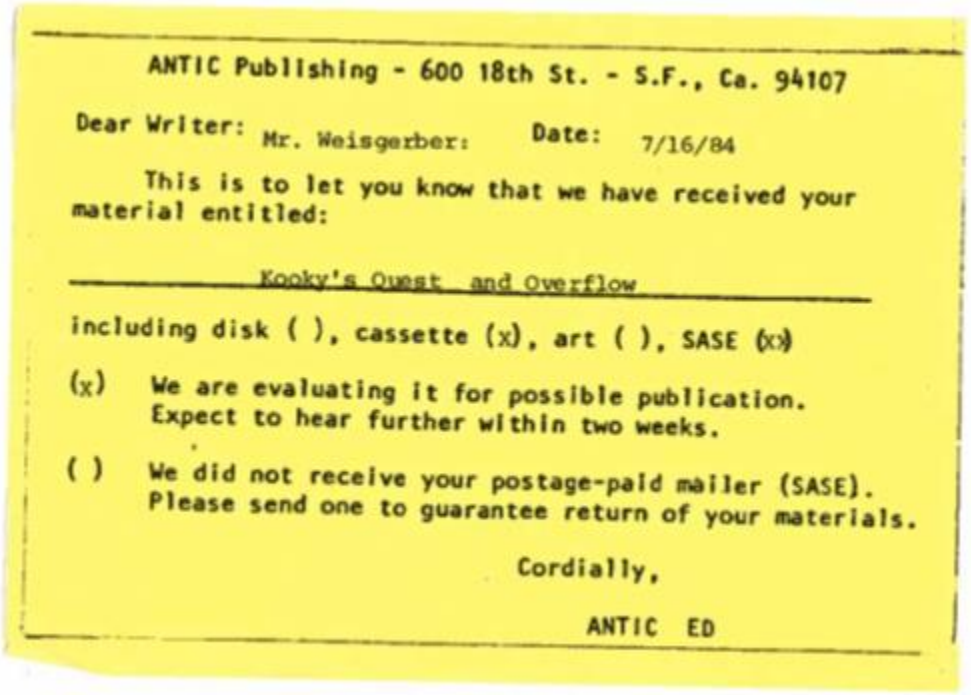
[redacted]

Richard Mansfield
Senior Editor

RM/jk

324 West Wendover Avenue, Suite 200. Greensboro, North Carolina 27408 USA.
Post Office Box 5406. Greensboro, North Carolina 27403 USA. Telephone 919-275-9809.
Telex Number 574411 WUPUBTLX GBO

# OVERFLOW



ANTIC Publishing - 600 18th St. - S.F., Ca. 94107

Dear Writer: Mr. Weisgerber: Date: 7/16/84

This is to let you know that we have received your material entitled:

Kooky's Quest and Overflow

including disk ( ), cassette (x), art ( ), SASE (x)

(x) We are evaluating it for possible publication. Expect to hear further within two weeks.

( ) We did not receive your postage-paid mailer (SASE). Please send one to guarantee return of your materials.

Cordially,

ANTIC ED

COMPUTE!™ Publications, Inc. abc
One of the ABC Publishing Companies

Re: Overflow

Dear Author:

We, at COMPUTE!, regret the delay in responding to your submission. However, after reviewing it we feel that it does not meet our present needs.

We are always interested, though, in reviewing programs and articles for possible publication. Please find enclosed a copy of COMPUTE!'s Author Guide which will assist you in submitting your material. Please do not hesitate to send us something in the future.

Thank you for thinking of COMPUTE!.

Yours truly,



Dan Carmichael
Submissions Editor

DC/vj

324 West Wendover Avenue, Suite 200. Greensboro, North Carolina 27408 USA.
Post Office Box 5406. Greensboro, North Carolina 27403 USA. Telephone 919-275-9809.



The ATARI® Resource
**ANTIC® Publishing Inc.**
524 Second Street
San Francisco, CA 94107
415 957 0886

July 17, 1984

John Weisgerber
[redacted]

Dear Mr. Weisgerber:

Thank you for submitting your program "Overflow" to Antic. All of us here at Antic agree that it's a great example of how much you can do in a very short program. Unfortunately, it's too hard to play. We always get wiped out in a matter of seconds, and as you can imagine, we're not exactly amateurs at playing these games. So if you would, please make the game a bit easier for us oldsters, and increment the difficulty as playing time increases. Thanks again.

Cordially,

[redacted]

David Duberman
Technical Editor

The ATARI® Resource
**ANTIC® Publishing Inc.**
524 Second Street
San Francisco, CA 94107
415 957 0886

John Weisgerber
[redacted]

Date: 8/3/84

Thank you for your recently submitted material entitled:

Overflow including

☐ listings, ☐ disk, ☒ cassette, ☐ art, ☒ SASE

☒ Your material has been accepted for publication in our _____ department, or as a feature now planned for the _____ issue, or for other use within one year. (X)

Payment will be made upon publication. If we do not use this within one year, we will pay a "kill fee" and copyright to the material reverts to you. If we need a biographical statement or copyright agreement from you, a form is enclosed. Please return these to us as soon as possible.

---

☐ Sorry, we can't accept your material. Unfortunately, we don't have room to print all the good submissions we receive.

☐ Your material is being returned to you herewith, or
☐ is being held here pending your postage-paid mailer.

The volume of our editorial work prevents a more personal response. We appreciate your efforts and are glad that you submitted to Antic. Please keep us in mind in the future.

PS: Payment will be made at the end of the Publication Month.

Cordially,
[redacted]
Nat Friedland,
Editor

( ) Enc.

# PIRACY II

Robert & Eric Arsenault
and John Weisenberger
[redacted]

Dear Sirs -

Enclosed is an article, listing, and both a disk and cassette save of an excellent arcade style game entitled Piracy II. This game is, in our opinion, one of the best arcade style games ever written in BASIC for the Atari. We are confident the game will be very marketable and hope to hear of your acceptance of it very soon. Thank you for considering our article.

Sincerely, one of the authors,
[redacted]

Instructions for loading and Running

Turn on disk drive. Insert PIRACY II disk. Turn on computer. Issue command RUN"D:PIRACY.II" After pressing start, there will be a long delay for initialization purposes. Refer to the article for instructions.


WORCESTER MASS
JUN 21 '84
U.S. POSTAGE
0.37
THE MAGAZINE FOR ATARI COMPUTER OWNERS
ANALOG
COMPUTING
565 MAIN STREET CHERRY VALLEY, MA 01611
ROBERT+ERIC ANSCHUETZ
AND JOHN WEISGERBER

PIRACY II 6-21-84

Dear contributor,

Thank you for your submission to ANALOG Computing. Our technical staff and editors have evaluated your article, but we find that it does not meet our needs at the present time. There are several reasons why we may not be publishing your article:

___ 1. We have already purchased a similar article, or have one under development.

X 2. The program/documentation needs improvement:

THE INITIALIZATION IS TOO SLOW ON RELPAYS!
CHARACTER SET SHOULD NOT HAVE TO BE LOADED EVERY TIME!
OTHER THAN THAT, IT LOOKS GREAT!!

___ 3. The article has limited appeal to our readers.

___ 4. The article needs re-writing:

___ 5. The article is not appropriate to our needs.

Thank you for submitting your article/ideas to ANALOG Computing.

Michael DesChenes
Publisher
ANALOG Computing

P.S.
NEXT TIME, YOU CAN SEND JUST THE DISK.

# ROBOT DUNGEON




ANTIC Publishing - 600 18th St. - S.F., Ca. 94107

Dear Writer: The Anschuetzs & Co.: Date: 6/21/84

This is to let you know that we have received your material entitled:

Robot Dungeon

including disk (X), cassette (X), art ( ), SASE ( )

( ) We are evaluating it for possible publication. Expect to hear further within two weeks.

(X) We did not receive your postage-paid mailer (SASE). Please send one to guarantee return of your materials

Cordially,

ANTIC ED


ANTIC PUBLISHING
524 - SECOND ST.
SAN FRANCISCO, CA. 94107
SAN FRANCISCO
JUL 10 84
1984
CALIF
U.S. POSTAGE
=020
METER
8203149




Robert & Eric Anschuetz
(John Weisgerber)
[redacted]

Date: 7/9/84

Thank you for your recently submitted material entitled:

Robot Dungeon

including ( ) listing(s), (X) disk(s), ( 2) cassette, ( ) art ( ) SASE

(XX) We are pleased to advise you that your material has been accepted

for the _____ department, or as a feature (XX), and is scheduled

for the _____ issue, or other use ( XX) within one year.

( ) Enclosed is our check # _____ for $ _____ as an advance pending publication. Final payment, if any is due, will be made then.
( X) Payment will be made on publication.

( ) We're sorry, your material does not meet our current editorial needs

( ) and is being returned herewith.

( ) and is being held pending receipt of a self-addressed, postage-paid mailer.

(X ) Enclosed are: ( ) Author's Guide, (X ) Copyright Agreement,
( ) Biographical Statement
Please return indicated items prior to publication.

The volume of our editorial work precludes a more personal response. We appreciate the time and effort you have invested in your material and are glad you submitted to ANTIC. Keep us in mind in the future.

Cordially,

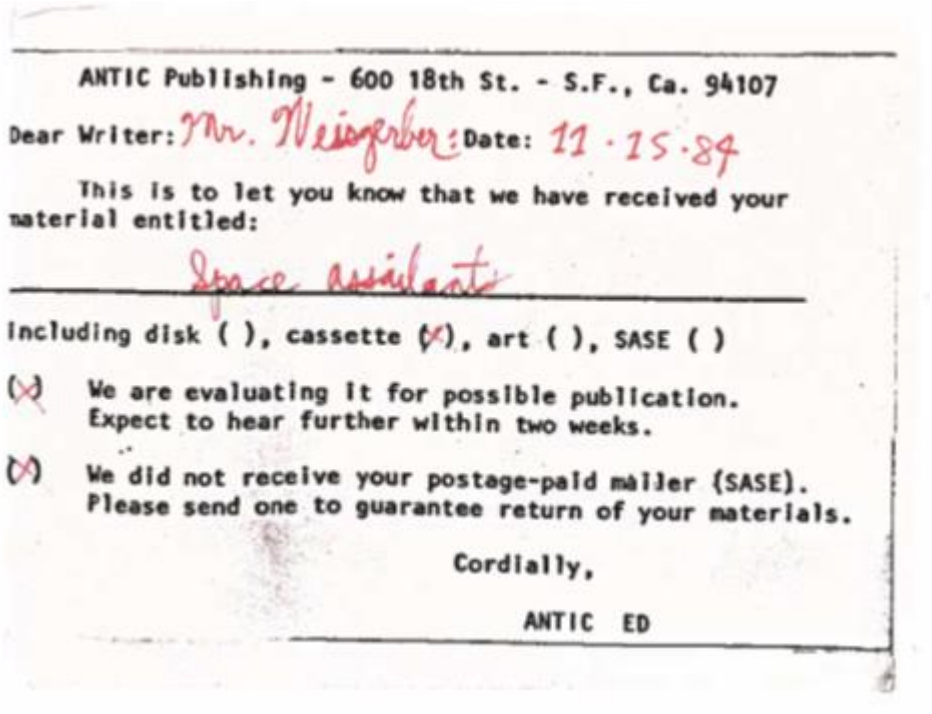
[redacted]

Robert DeWitt,
Managing Editor

( ) Enc.

Rev. 9/16/83

# SPACE ASSAILANTS



ANTIC Publishing - 600 18th St. - S.F., Ca. 94107

Dear Writer: Mr. Weisgerber: Date: 11·15·84

This is to let you know that we have received your material entitled:

Space assailants

including disk ( ), cassette (X), art ( ), SASE ( )

(X) We are evaluating it for possible publication. Expect to hear further within two weeks.

(X) We did not receive your postage-paid mailer (SASE). Please send one to guarantee return of your materials.

Cordially,

ANTIC ED

COMPUTE!™ Publications, Inc. abc
One of the ABC Publishing Companies

October 9, 1984

Mr. John Weisgerber
[redacted]

Dear Mr. Weisgerber:

In order to facilitate our review of your program, "Space Assailants," please send us another tape. We discovered that this tape will not load.

Please write "PENDING" in the lower left hand corner of the envelope and send it to the address below.

Thank you for your interest in COMPUTE!.

Yours truly,

[redacted]

Mark Tuttle
Submissions Editor

MT/jbf

Enclosure

324 West Wendover Avenue, Suite 200. Greensboro, North Carolina 27408 USA.
Post Office Box 5406. Greensboro, North Carolina 27403 USA. Telephone 919-275-9809.
Telex Number 574411 WUPUBTLX GBO

COMPUTE!™ Publications, Inc. abc
One of the ABC Publishing Companies

Re: "Space Assailants"

Dear Author:

We, at COMPUTE!, regret the delay in responding to your submission. However, after reviewing it we feel that it does not meet our present needs.

We are a young magazine, growing at a very rapid rate. This has, unfortunately, resulted in some delays in our editorial review process. We have recently reorganized and expect that editorial evaluations in the future will be prompt.

We are always interested, though, in reviewing programs and articles for possible publication. Please find enclosed a copy of COMPUTE!'s Author Guide which will assist you in submitting your material. Please do not hesitate to send us something in the future.

Thank you for thinking of COMPUTE!.

Yours truly,

[redacted]

Dan Carmichael
Submissions Editor

DC/vj

505 Edwardia Drive, Greensboro, North Carolina 27409 USA
Post Office Box 5406, Greensboro, North Carolina 27403 USA, Telephone 919-275-9809.

# COMPUTE!'s Author Guide

Most of the following suggestions serve to improve the speed and accuracy of publication. **COMPUTE!** is primarily interested in new and timely articles on VIC, Apple, PET/CBM, Commodore 64, Atari, Sinclair/Timex, TI-99/4A, and Radio Shack Color Computer. We are much more concerned with the content of an article than with its style. Above all, articles should be clear and well-explained.

The guidelines below will permit your good ideas and programs to be more easily edited and published:

**1.** The upper left corner of the first page should contain your name, address, telephone number, and the date of submission.

**2.** The following information should appear in the upper right corner of the first page. If your article is specifically directed to one make of computer, please state the brand name and, if applicable, the BASIC or ROM or DOS version(s) involved. In addition, please indicate the memory requirements of programs.

**3.** The underlined title of the article should start about 2/3 of the way down the first page.

**4.** Following pages should be typed normally, except that in the upper right corner there should be an abbreviation of the title, your last name, and the page number. For example: Memory Map/Smith/2.

**5.** Short programs (under 20 lines) can easily be included within the text. Longer programs should be separate listings. *It is essential that we have a copy of the program, recorded twice, on a tape or disk.* The tape or disk should be labeled with the author's name, the title of the article, and, if applicable, the BASIC/ROM/DOS version(s). Atari tapes should specify whether they are to be LOADed or ENTERed. We prefer to receive Apple programs on disk rather than tape. On the other hand, tapes are preferred for the TI and Radio Shack computers. Tapes are fairly sturdy, but disks need to be enclosed within plastic or cardboard mailers (available at photography, stationery, or computer supply stores).

It is far easier for others to type in your program if you use CHR$(X) values and TAB(X) or SPC(X) instead of cursor manipulations to format your output. For five carriage returns, FOR I=1 TO 5:PRINT:NEXT is far more "portable" to other computers with other BASICs and also easier to type in. And, instead of a dozen right-cursor symbols, why not simply use PRINT SPC(12)? A quick check through your program – making these substitutions – would be greatly appreciated by your editors and by your readers.

**6.** If your article is accepted and you have since made improvements to the program, please submit an entirely new tape or disk and a new copy of the article reflecting the update. We cannot easily make revisions to programs and articles. It is necessary that you send the revised version as if it were a new submission entirely, but be sure to indicate that your submission is a revised version by writing "Revision" on the envelope and the article.

**7.** All lines within the text of the article should be spaced so that there is about 1/2 inch between them. A one inch margin should be left at the right, left, top, and bottom of each page. No hyphens should be used at the ends of lines to break words. And please do not justify. Leave the lines ragged.

**8.** Standard typing paper should be used (no onionskin or other thin paper) and typing should be on one side of the paper only (upper- and lowercase).

**9.** Sheets should be attached together with a paper clip. Staples should not be used.

**10.** A good general rule is to spell out the numbers zero through ten in your article and write higher numbers as numerals (1024). The exceptions to this are: Figure 5, Table 3, TAB(4), etc. Within ordinary text, however, the zero through ten should appear as words, not numbers. Also, symbols and abbreviations should not be used within text: use "and" (not &), "reference" (not ref.), "through" (not thru).

**11.** For greater clarity, use all capitals when referring to keys (RETURN, TAB, ESC, SHIFT), BASIC words (LIST, RND, GOTO), and three languages (BASIC, APL, PILOT). Headlines and subheads should, however, be initial caps only, and emphasized words are not capitalized. If you wish to emphasize, underline the word and it will be italicized during typesetting.

**12.** **COMPUTE!** pays between $75 and $600 for published articles. In general, the rate reflects the length of the article. Payment is made upon acceptance of an article. Following submission (Editorial Department, **COMPUTE!** Magazine, P.O. Box 5406, Greensboro, NC 27403) it will take from two to four weeks for us to reply. If your work is accepted, you will be notified by a letter which will include a contract for you to sign and return. Rejected manuscripts are returned to authors who enclose an SASE. We do not consider articles which are multiple submissions. If you wish to send an article to another magazine for consideration, please do not submit it to us.

**13.** Articles can be of any length – from a single-line routine to a multi-issue series. The average article is about four to eight double-spaced, typed pages.

**14.** If you want to include photographs, they should be 5x7, black and white glossies.

THE MAGAZINE FOR ATARI COMPUTER OWNERS

ANALOG COMPUTING

SPACE ASSAILANTS — ARTICLE TITLE

7-31-84 — DATE

Dear contributor,

Thank you for your submission to **ANALOG Computing**. Our technical staff and editors have evaluated your article, but we find that it does not meet our needs at the present time. There are several reasons why we may not be publishing your article:

☐ 1. We have already purchased a similar article, or have one under development.

☐ 2. The program/documentation needs improvement:

☒ 3. The article has limited appeal to our readers.

☐ 4. The article needs re-writing:

☐ 5. The article is not appropriate to our needs.

Thank you for submitting your article/ideas to **ANALOG Computing**.

AUTHORIZED SIGNATURE

Lee H. Pappas
Publisher, **ANALOG Computing**

[redacted]

AUTHORIZED SIGNATURE

Charles Bachand
Technical/Software, **ANALOG Computing**

Additional comments:

SORRY, BUT WE HAVE ENOUGH "SPACE "SHOOT 'EM UP'S" TO HOLD US FOR TWO YEARS! WISH THERE WAS A WAY TO DO THE SAME THING AND NOT HAVE IT FLICKER.

AGAIN, SORRY...

The ATARI® Resource
**ANTIC® Publishing Inc.**
524 Second Street
San Francisco, CA 94107
415 957 0886

John Weisgerber
[redacted]

Date: 11/20/84

Thank you for your recently submitted material entitled:

Space Assailants including

☒ listings, ☐ disk, ☒ cassette, ☐ art, ☐ SASE

☐ Your material has been accepted for publication in our _____ department, or as a _____ now planned for the _____ issue, or for other use within one year.

Payment will be made upon publication. If we do not use this within one year, we will pay a "kill fee" and copyright to the material reverts to you. If we need a biographical statement or copyright agreement from you, a form is enclosed. Please return these to us as soon as possible.

---

☒ Sorry, we can't accept your material. Unfortunately, we don't have room to print all the good submissions we receive.

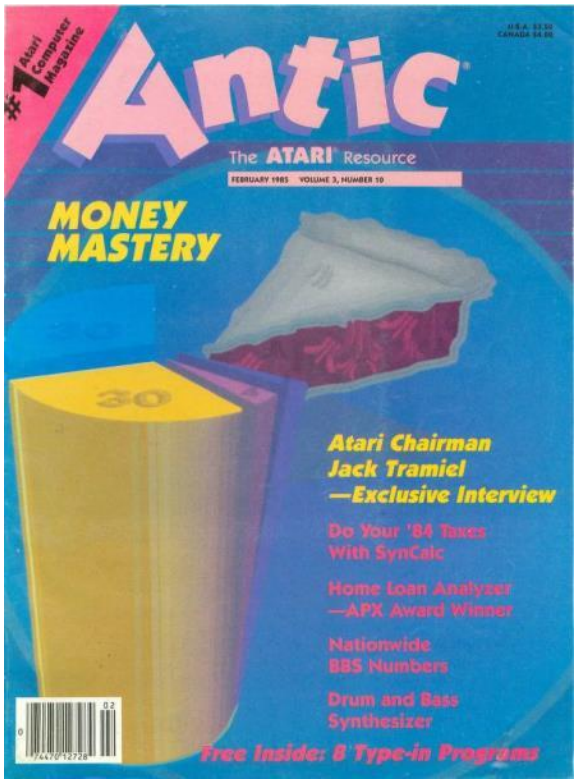
☐ Your material is being returned to you herewith, or
☒ is being held here pending your postage-paid mailer.

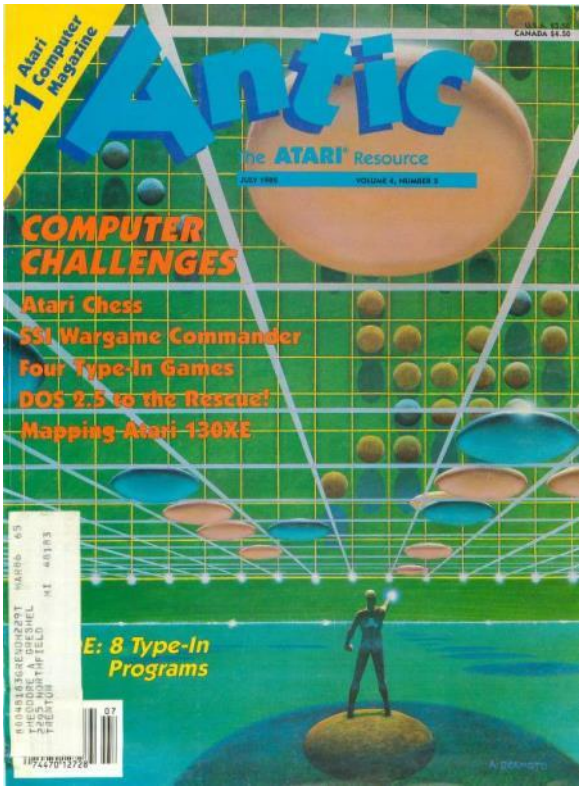
The volume of our editorial work prevents a more personal response. We appreciate your efforts and are glad that you submitted to Antic. Please keep us in mind in the future.

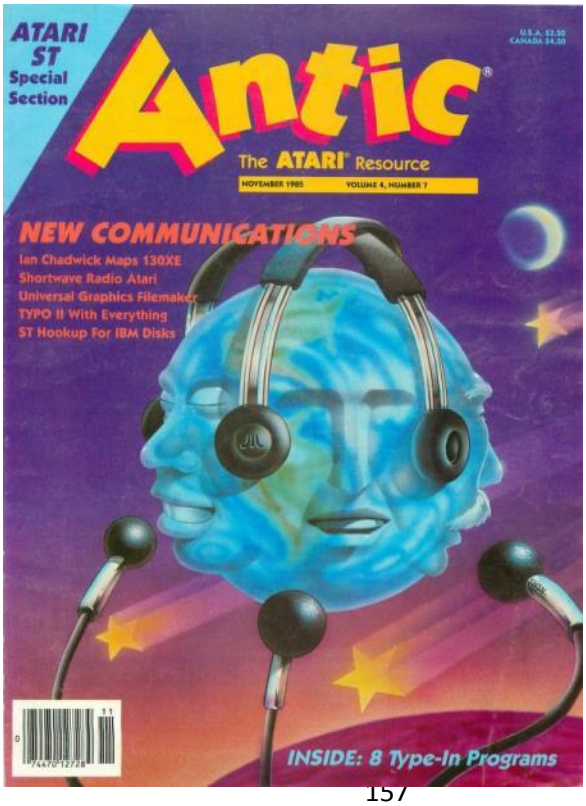
Cordially,
[redacted]
Nat Friedland,
Editor

( ) Enc.

# A/W/A PUBLISHED GAMES

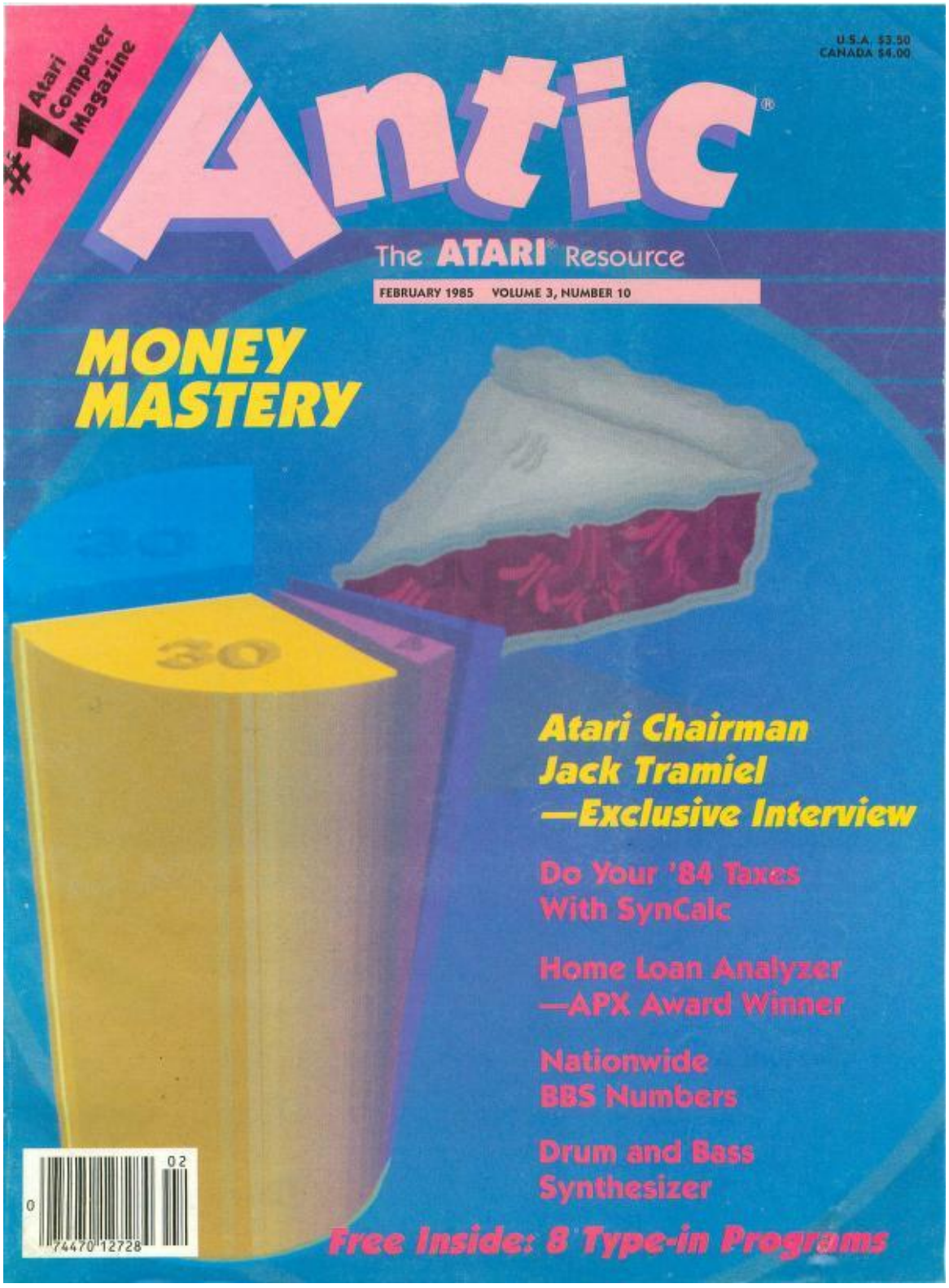


# KOOKY'S QUEST

**ANTIC VOL. 3, NO. 10 / FEBRUARY 1985**





3 enhanced 800XL's in '85?.....72



Take apart your home loan .....18



New sounds from your Atari .....26



Kooky's Quest — Not for the squeamish.....14

# Antic®

The ATARI® Resource FEBRUARY 1985, VOLUME 3, NUMBER 10

## FEATURES

**SYNTEGRATED SOFTWARE** by Michael Ciraolo 14
Finally...compatible business applications for Atari

**HOME LOAN ANALYZER** by Jim Skinner 18
APX 1983 prize winner — now in Antic! TYPE-IN SOFTWARE

**INCOME TAX SPREADSHEET** by K.W. Harms 22
Preparing tax returns on Atari TYPE-IN SOFTWARE

**DRUM SYNTH / BASS SYNTH** by Glenn Gutierrez 26
Powerful AUDCTL rhythm section TYPE-IN SOFTWARE

**CURRENCY CHANGER** by J. Goolevitch and W. Wakeman 31
Convert your drachmas to pesos TYPE-IN SOFTWARE

**BEAT THE BEEPS** by Ian Lovejoy 33
Turn off annoying keyboard sound TYPE-IN SOFTWARE

**JACK TRAMIEL DECLARES WAR!** 72
Antic's Exclusive Interview With The Atari Chairman

## DEPARTMENTS

**GAME OF THE MONTH** TYPE-IN SOFTWARE
KOOKY'S QUEST by Anshuetz, Anshuetz and Weisgerber 56

**COMMUNICATIONS**
200+ ATARI BULLETIN BOARDS 59

**ASSEMBLY LANGUAGE** TYPE-IN SOFTWARE
S.A.M. HANDLER by Chris Bone 63

**EDUCATION** TYPE-IN SOFTWARE
MUSIC FLASHCARDS by James Brown 66

**TOOLBOX**
PARALLEL BUS REVEALED: Part II by Earl Rice 68

## SOFTWARE LIBRARY

**TYPE-IN LISTINGS SECTION** 35

EDITORIAL 6
I/O BOARD 10
HELP! 12
PRODUCT REVIEWS 74
ATARI SERVICE CENTERS 78
ADVERTISER'S LIST 80
SHOPPER'S GUIDE 81

game of the month

# KOOKY'S QUEST

Article on page 56.

```
EH 10 REM KOOKY'S QUEST
IS 20 REM BY
WZ 30 REM ANSCHUETZ/WEISGERBER/ANSCHUETZ
RI 40 REM ANTIC MAGAZINE
EK 100 GRAPHICS 18:POKE 16,64:POKE 53774,
   64:DL=PEEK(560)+256*PEEK(561):SETCOLOR
    2,8,0
CL 110 POKE DL+10,6:POKE DL+11,6:POKE DL+
   12,6:POKE DL+13,2:SETCOLOR 0,8,4:POSIT
   ION 4,4
WH 120 ? #6;"KOOKY'S QUEST":POSITION 5,8:
   ? #6;"ANSCHUETZ/WEISGERBER/ANSCHUETZ"
OD 130 POSITION 8,6:? #6;"[ESC] start [CTRL
   1]":IF PEEK(53279)<>6 THEN 130
HP 140 DIM A$(25),B$(25):GRAPHICS 8:POKE
   16,64:POKE 53774,64:POKE 752,1:SETCOLO
   R 2,0,2:SETCOLOR 4,8,2
OH 150 A$="<++<++<++ ++ ++ ++<++<++<":B$=
   " ++ ++ ++ ++ ++ ++ ++ ++ ++ "
SL 160 DIM OHP(6,6),MON$(6),CON$(6),T$(2)
   ,WEP$(12):MON$="^;(/:\":CON$="[      ":
   T$="J_":WEP$="ghefcdabijkl"
SJ 170 FOR R=1 TO 6:FOR C=1 TO 6:READ A:O
   HP(R,C)=A:NEXT C:NEXT R
SZ 180 DIM F$(16):F$="<++<++<++<++<++<++<"
SZ 190 DIM G$(25),H$(25):G$="<++<++<++<++
   <++<++<++<++<++<":H$="<++ ++ ++ ++ ++ ++
   ++ ++<"
DI 200 DATA 30,15,10,6,5,7,5,10,30,15,5,7
   ,5,6,15,10,30,7,5,6,5,5,7,5,6,15,30,10
   ,7,5,6,5,10,15,30,6,5,7,5,30,10,15
NI 210 DATA 0,0,64,248,64,0,0,0
DB 220 DATA 0,0,2,31,2,0,0,0
OJ 230 DATA 0,27,10,254,10,27,0,0
MS 240 DATA 0,216,80,127,80,216,0,0
OY 250 DATA 0,31,14,254,14,31,0,0

JE 260 DATA 0,248,112,127,112,248,0,0
NA 270 DATA 0,64,96,255,96,64,0,0
MH 280 DATA 0,2,6,255,6,2,0,0
GS 290 DATA 0,0,63,24,252,0,0,0
GI 300 DATA 0,0,252,24,63,0,0,0
ED 310 DATA 0,40,18,92,58,72,20,0
FD 320 DATA 0,20,72,58,92,18,40,0
ED 330 DATA 255,135,219,219,219,219,135,2
   55
RV 340 DATA 126,255,255,231,231,255,255,1
   26
SK 350 DATA 60,24,36,66,66,36,24,0
KZ 360 DATA 85,85,85,85,85,85,85,85
MD 370 DATA 254,254,254,0,239,239,239,0
OY 380 DATA 0,126,219,255,189,195,126,0
PB 390 DATA 255,231,195,195,231,231,231,2
   55
LB 400 DATA 255,219,147,219,219,219,193,2
   55
DM 410 DATA 126,255,255,255,255,255,255,1
   26
NB 420 DATA 73,114,149,12,126,146,34,85
OU 430 DATA 28,62,42,28,42,34,85,85
HM 440 DATA 36,219,126,66,255,165,36,231
VY 450 DATA 0,0,219,126,24,36,0,0
RK 460 DATA 240,80,248,30,7,33,67,62
SI 470 DATA 0,28,42,28,73,62,8,54
DV 480 DATA 124,84,124,48,254,56,124,238
TM 490 DATA 0,0,192,224,191,234,192,0
MP 500 DATA 60,66,129,255,165,165,255,0
DD 510 DATA 170,170,170,170,170,170,170,1
   70
RG 520 ? "K"
BB 530 DIM CHA$(31):CHA$="abcdefghijkl=+_
   @<>x&#[^;:/\()]?":CH$=(PEEK(106)-8)*25
   6
SW 540 PM=(PEEK(106)-12)*256:IF PEEK(CH$+
   9)<>0 THEN 590
TB 545 ? :? :? "    One moment please..."
   :FOR X=1 TO 700:NEXT X:? "K":POKE 559,
   0
LH 550 FOR I=0 TO 1023:POKE CH$+I,PEEK(57
   344+I):NEXT I:FOR I=1 TO 31:CHP=CH$+(A
   SC(CHA$(I))-32)*8
PZ 560 IF I<13 THEN CHP=CHP+256
JO 570 FOR J=0 TO 7:READ A:POKE CHP+J,A:N
   EXT J:NEXT I
TL 580 FOR X=PM+64 TO PM+128:POKE X,255:N
   EXT X
PO 590 POKE 53277,3:POKE 54279,PM/256-4:P
   OKE 623,1:POKE 53256,3:POSITION 2,12:P
   OKE 559,62
BL 600 OPEN #1,4,0,"K:":? "++++SKILL - [EX
   PERT OR [BEGINNER":;GET #1,K:IF K<>69 A
   ND K<>66 THEN CLOSE #1:? "+":GOTO 600
BE 610 ? "K":POKE 559,0:POKE 704,2:P=188:
   SC=PEEK(88)+256*PEEK(89):L=1:X=30:X1=S
   C+472:LI=10:RL=7
DF 620 MO1=4:MO2=12:O=1:ST=23:TS=15:POKE
   82,0
TS 630 POKE 756,CH$/256:GOSUB 1070
UX 640 POSITION 4,3:? "KOOKY'S+++++++QUES
   T":POSITION 1,6:? "SCORE=0+++++++++LIV
   ES=10+++++++++LEVEL=1"
PP 650 POSITION 1,12:? "TREASURES=0"
TR 660 FOR Y=18 TO 21:POSITION 0,Y:? "<<<
   <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<"
   ;:NEXT Y
YD 670 POSITION 27,3:? "??????????":POSIT
   ION 27,12:? "??????????"
GA 680 FOR R=4 TO 11:POSITION 27,R:? "?":
   POSITION 36,R:? "?":NEXT R
EJ 690 POSITION 5,19:? "1 SWORD   3 MACE
      5 L.BOLT  ":POSITION 5,20:? "2 AXE
      4 KNIFE   6 FIREBALL"
RR 700 POSITION 14,3:? "eeeeeeeeeee":POSI
   TION 14,13:? "eeeeeeeeeee":FOR R=4 TO
   12:POSITION 14,R:? "e":POSITION 24,R
DJ 710 ? "e":NEXT R:FOR R=4 TO 6:POSITION
    15,R:? "<<<    <<<":POSITION 15,R+6:?
   "<<<<<<<<<<":NEXT R
NB 720 N=PEEK(X1):POKE X1,N+128:POSITION
   2,0:? " ":POKE 704,42*(K=69)+132*(K=66
   ):POKE 559,62:POKE 53248,160
GI 730 J=STICK(0)
AU 740 IF  NOT (CR=RL AND ((DMT<5 AND L>1
   ) OR (DMT<1 AND L=1))) THEN 760
PO 750 POSITION 4,16:? "YOU WILL WANDER L
   EVEL ";L;" FOREVER":FOR MIAD=0 TO 0 ST
   EP 0:NEXT MIAD
OC 760 IF J=14 AND (N=65 OR N=67 OR N=68
   OR N=88 OR N=90 OR N=83 OR N=124 OR N<
   11) THEN ST=1:A=7:A1=9:GOTO 820
JD 770 IF J=13 AND (N=65 OR N=81 OR N=68
   OR N=69 OR N=87 OR N=83 OR N=124 OR N<
   11) THEN ST=-1:A=9:A1=7:GOTO 820
DP 780 IF J=7 AND (N=65 OR N=81 OR N=87 O
   R N=82 OR N=88 OR N=90 OR N=83) THEN S
   T=-1:A=20:A1=18:GOTO 1020
```

continued on next page

```
GW 790 IF J=11 AND (N=67 OR N=68 OR N=69
   OR N=87 OR N=82 OR N=88 OR N=83) THEN
   ST=1:A=18:A1=20:GOTO 1020
ZZ 800 IF N=6 OR N=10 THEN 2060
PE 810 GOTO 730
TX 820 V=ST*3:POKE X1,N
XM 830 IF N=124 OR N=6 OR N=10 THEN 850
LN 840 GOSUB 950*(N=69 OR N=68 OR N=67)+9
   80*(N=65 OR N=90 OR N=81)+1000*(N=83 O
   R N=87 OR N=88)
VF 850 A=A-V:A1=A1-V:X1=X1-ST*40:N=PEEK(X
   1):POKE X1,N+128
LQ 860 IF N=65 THEN 1890
HO 870 IF N=68 THEN 1910
AO 880 IF N=67 OR N=69 THEN 1930
EF 890 IF N=87 OR N=88 THEN 1950
YA 900 IF N=81 OR N=90 THEN 1970
LY 910 IF N=83 THEN 1990
UN 920 IF N=3 OR N=11 THEN 2010
AG 930 IF N=6 OR N=10 THEN 2060
XL 940 POKE 77,0:GOTO 730
HK 950 FOR W=A TO A1 STEP ST:POSITION 15,
   W:? "<<<":POSITION 15,W+V:? "       ":N
   EXT W
UH 960 FOR W=A TO A1 STEP ST:POSITION 15,
   W+V:? "<<<":NEXT W:RETURN
PR 970 GOTO 730
KD 980 FOR W=A TO A1 STEP ST:POSITION 21,
   W:? "<<<":POSITION 18,W+V:? "       ":N
   EXT W
OF 990 FOR W=A TO A1 STEP ST:POSITION 21,
   W+V:? "<<<":NEXT W:RETURN
IH 1000 FOR W=A TO A1 STEP ST:POSITION 15
   ,W:? "<<<   <<<":POSITION 15,W+V:? "
         ":NEXT W
YC 1010 FOR W=A TO A1 STEP ST:POSITION 15
   ,W+V:? "<<<   <<<":NEXT W:RETURN
RB 1020 V=ST*3:POKE X1,N
HF 1030 DO=2050*(N=65 OR N=68 OR N=83)+21
   00*(N=67 OR N=88 OR N=90)+2150*(N=69 O
   R N=87 OR N=81)
JM 1040 IF N<>82 THEN GOSUB 1140*(N=65 OR
    N=68 OR N=83)+1160*(N=67 OR N=88 OR N
   =90)+1180*(N=69 OR N=87 OR N=81)
SW 1050 A=A-V:A1=A1-V:X1=X1-ST:N=PEEK(X1)
   :POKE X1,N+128
ZI 1060 IF N=88 THEN 2130
YT 1070 IF N=83 THEN 2150
HD 1080 IF N=90 OR N=67 THEN 2170
LH 1090 IF N=65 OR N=68 THEN 2190
AK 1100 IF N=69 OR N=81 THEN 2210
ZD 1110 IF N=87 THEN 2230
KU 1120 IF N=3 OR N=11 THEN 2250
VY 1130 POKE 77,0:GOTO 730
NP 1140 FOR W=A TO A1 STEP ST:POSITION W+
   V,4:? B$:POSITION W,4:? A$:NEXT W
GZ 1150 FOR W=A TO A1 STEP ST:POSITION W+
   V,4:? A$:NEXT W:RETURN
DC 1160 FOR W=A TO A1 STEP ST:POSITION W+
   V,4:? B$(10):POSITION W,4:? G$(19):NEX
   T W
FS 1170 FOR W=A TO A1 STEP ST:POSITION W+
   V,4:? G$(19):NEXT W:RETURN
ZF 1180 FOR W=A TO A1 STEP ST:POSITION W+
   V,7:? B$(10):POSITION W,10:? G$(19):NE
   XT W
FS 1190 FOR W=A TO A1 STEP ST:POSITION W+
   V,10:? G$(19):NEXT W:RETURN
GG 1200 OP=0:DM=0:GOTO 1210
OC 1210 Z=INT(RND(0)*2):M3=16+6*Z:MM3=8:D
   M=0:MON=INT(RND(1)*5)+2:KX=19:KY=8:THP
   =0:IF L=5 THEN MON=1
WP 1220 TX=INT(RND(0)*7)+16:TY=INT(RND(0)
   *7)+5:TY=TY+(TY=8):POSITION TX,TY:? T$
   (Z*1,Z+1)
AM 1230 IF L=1 AND X1=SC+428 THEN POSITIO
   N 19,5:? "X"
KT 1240 IF L=5 AND X1=SC+195 THEN POSITIO
   N 19,5:? "J"
XI 1250 POSITION M3,MM3:? MON$(MON,MON):P
   OSITION KX,KY:? ">"
OA 1260 POSITION 10,16:? "CHOOSE YOUR WEA
   PON!":GET #1,OWEP:OWEP=OWEP-48:POSITI
   ON 10,16:? "                    "
IL 1270 IF OWEP>6 OR OWEP<1 THEN 1260
AI 1280 MIAD=1^1^1:HP=OHP(MON,OWEP):OWEP=
   OWEP*2-1
SW 1290 POSITION M3,MM3:? CON$(MON,MON):B
   C=INT(2*RND(0)+1):CB=INT(2*RND(0)+1)
RX 1300 POSITION M3,MM3:? CON$(MON,MON):B
   C=INT(2*RND(0)+1):CB=INT(2*RND(0)+1)
RJ 1310 M3=M3+(BC=1)*(M3<22)-(BC=2)*(M3>1
   6)
LE 1320 MM3=MM3+(CB=1)*(MM3<11)-(CB=2)*(M
   M3>5)
EM 1330 POSITION WX,KY:? " ":WX=WX*JW
ID 1340 POSITION M3,MM3:? MON$(MON,MON)
OD 1350 J=STICK(0)
UO 1360 IF STRIG(0)=0 OR JW=1 THEN 1480
KX 1370 IF J=14 THEN POSITION KX,KY:? " "
   :KY=KY-1
HC 1380 IF J=13 THEN POSITION KX,KY:? " "
   :KY=KY+1
CT 1390 IF J=7 THEN POSITION KX,KY:? " ":
   KX=KX+1
FM 1400 IF J=11 THEN POSITION KX,KY:? " "
   :KX=KX-1
CD 1410 PE=PEEK(SC+KX+KY*40):IF PE=61 OR
   PE=63 THEN SOUND 0,18,10,10:DMT=DMT+1^
   1:SOUND 0,0,0,0:PT=PT+5:GOTO 1460
FS 1420 IF PE=9 THEN SOUND 0,18,10,10:KEY
   =1^1:SOUND 0,0,0,0:GOTO 1460
MW 1430 IF PE=5 AND KEY THEN POSITION 4,1
   6:? "YOU HAVE ESCAPED THE DUNGEON!!!":
   FOR MIAD=0 TO 0 STEP 0:NEXT MIAD
HX 1440 IF PE=5 THEN 1460
CW 1450 IF PE>0 AND PE<>38 THEN SOUND 0,1
   0,100,10:LI=LI-1:GOTO 1660
DV 1460 POSITION KX,KY:? ">"
TC 1470 GOTO 1290
OO 1480 J=STICK(0)
GX 1490 WEP=OWEP:PE=PEEK(SC+KX+KY*40):IF
   PE>0 AND PE<>38 THEN SOUND 0,10,100,10
   :LI=LI-1:GOTO 1660
DV 1500 IF OWEP>6 THEN 1550
XB 1510 JW=0
PW 1520 IF J=7 AND KX<22 THEN WX=KX+1:GOT
   O 1620
JJ 1530 IF J=11 AND KX>16 THEN WEP=WEP+1:
   WX=KX-1:GOTO 1620
SV 1540 GOTO 1290
OH 1550 IF JW=1 THEN 1590
NG 1560 IF J=7 AND KX<22 THEN RA=1:WX=KX:
   GOTO 1610
PJ 1570 IF J=11 AND KX>16 THEN RA=-1:WX=K
   X:GOTO 1610
TH 1580 GOTO 1290
ZX 1590 PE1=PEEK(SC+WX+KY*40):IF PE1=61 O
   R PE1=63 OR PE1=59 THEN WX=0:JW=0:GOTO
    1290
EJ 1600 IF PE1>0 THEN SOUND 0,16,14,10:JW
   =0:THP=THP+HP*1^1:SOUND 0,0,0,0:IF THP
   =30 THEN PT=PT+10:GOTO 1660
EV 1610 WEP=WEP+(RA=-1):WX=WX+RA*(WX<23)*
   (WX>15):JW=(WX<22)*(WX>16)
ZE 1620 PE1=PEEK(SC+WX+KY*40):IF PE1=61 O
   R PE1=63 OR PE1=59 THEN WX=0:JW=0:GOTO
    1290
ES 1630 IF PE1>0 THEN SOUND 0,16,14,10:JW
   =0:THP=THP+HP*1^1:SOUND 0,0,0,0:IF THP
   =30 THEN PT=PT+10:GOTO 1660
XR 1640 POSITION WX,KY:? WEP$(WEP,WEP)
TA 1650 GOTO 1290
CN 1660 FOR W=5 TO 11:POSITION 16,W:? "
         ":NEXT W:POSITION 7,8:? LI;" ":SO
   UND 0,0,0,0:CR=CR+1:POSITION 7,6:? PT
KP 1670 POSITION 11,12:? DMT
FG 1680 IF LI<1 THEN POSITION 15,15:? "GA
```

```
   ME OVER":POSITION 7,16:? "PRESS BUTTON
    TO PLAY AGAIN":GOTO 2480
CY 1690 J=STICK(0):UP=PEEK(X1-40):DO=PEEK
   (X1+40):FO=PEEK(X1+1):BA=PEEK(X1-1)
QR 1700 IF J=14 AND (UP=65 OR UP=81 OR UP
   =68 OR UP=69 OR UP=87 OR UP=83 OR UP=1
   24) THEN OW=1:ST=1:GOTO 1750
DS 1710 IF J=13 AND (DO=65 OR DO=67 OR DO
   =68 OR DO=88 OR DO=90 OR DO=83 OR DO=1
   24) THEN OW=2:ST=-1:GOTO 1750
DA 1720 IF J=11 AND (BA=65 OR BA=81 OR BA
   =87 OR BA=82 OR BA=88 OR BA=90 OR BA=8
   3) THEN OW=3:ST=1:GOTO 1810
ZI 1730 IF J=7 AND (FO=67 OR FO=68 OR FO=
   69 OR FO=87 OR FO=82 OR FO=88 OR FO=83
   ) THEN OW=4:ST=-1:GOTO 1810
UV 1740 GOTO 1690
NS 1750 IF OW=1 THEN A=4:A1=11
SN 1760 IF OW=2 THEN A=12:A1=5
VO 1770 POSITION 16,A1+ST:? "        ":FOR
    W=A TO A1 STEP ST:POSITION 15,W:? "<<
   <    <<<":POSITION 15,W+ST
TJ 1780 ? "<<<<<<<<<<":NEXT W
BI 1790 POSITION 15,A1+ST:? "<<<    <<<"
AY 1800 V=3*ST:A=7*(OW=1)+9*(OW=2):A1=9*(
   OW=1)+7*(OW=2):POKE X1,11:GOTO 850
TN 1810 IF OW=3 THEN A=15:A1=22
WM 1820 IF OW=4 THEN A=23:A1=16
WC 1830 POSITION A1+ST,4:? #5
SA 1840 FOR W=A TO A1 STEP ST:POSITION W,
   4:? A$
HB 1850 POSITION W+ST,4:? G$:NEXT W:POSIT
   ION A1+ST,7:? B$(19)
RC 1860 V=3*ST:A=18*(OW=3)+20*(OW=4):A1=2
   0*(OW=3)+18*(OW=4):POKE X1,11:GOTO 105
   0
MU 1870 RESTORE 2300+(L-1)*20
KM 1880 FOR P=5C*100 TO 5C*468 STEP 40:FO
   R RT=P TO P+7:READ M:POKE RT,M:NEXT RT
   :NEXT P:RETURN
TC 1890 FOR W=A TO A1 STEP ST:POSITION 21
   ,W:? "   ":NEXT W
IV 1900 FOR W=A TO A1 STEP ST:POSITION 21
   ,W:? ">>>":POSITION 21,W+V:? "   ":NEX
   T W:GOTO 730
WR 1910 FOR W=A TO A1 STEP ST:POSITION 15
   ,W:? "   ":NEXT W
UL 1920 FOR W=A TO A1 STEP ST:POSITION 15
   ,W:? "<<<":POSITION 15,W+V:? "   ":NEX
   T W:GOTO 730
WX 1930 FOR W=A TO A1 STEP ST:POSITION 15
   ,W:? "   ":NEXT W
WD 1940 FOR W=A TO A1 STEP ST:POSITION 15
   ,W:? ">>>>>>>":POSITION 15,W+V:? "   ":
   NEXT W:GOTO 730
YJ 1950 FOR W=A TO A1 STEP ST:POSITION 15
   ,W:? "         ":NEXT W
GX 1960 FOR W=A TO A1 STEP ST:POSITION 15
   ,W:? ">>>>>>>>>>":POSITION 15,W+V:? "
        ":NEXT W:GOTO 730
SY 1970 FOR W=A TO A1 STEP ST:POSITION 21
   ,W:? "   ":NEXT W
WE 1980 FOR W=A TO A1 STEP ST:POSITION 18
   ,W:? ">>>>>>>":POSITION 21,W+V:? "   ":
   NEXT W:GOTO 730
YV 1990 FOR W=A TO A1 STEP ST:POSITION 15
   ,W:? "         ":NEXT W
DG 2000 FOR W=A TO A1 STEP ST:POSITION 15
   ,W:? ">>>   <<<":POSITION 15,W+V:? "
        ":NEXT W:GOTO 730
JZ 2010 POSITION 15,A:? ">>>>>>>>>>"
LO 2020 A1=11*(ST=1)+5*(ST=-1)
AY 2030 FOR W=A TO A1 STEP ST:POSITION 15
   ,W:? ">       <":POSITION 15,W+ST:? ">
   >>>>>>>>":NEXT W
CL 2040 POSITION 15,A:? ">>>>>>>>>>":IF N=
   3 THEN OP=0:DM=0:GOTO 1210
UL 2050 GOTO 1690
BD 2060 IF (L>1 AND DMT<5) OR (L=1 AND DM
   T<1) THEN 730
AB 2070 CR=0:PT=PT+50:IF N=6 THEN L=L-1:D
   MT=0:N=10:GOTO 2100
GK 2080 IF N=10 THEN L=L+1:DMT=0:N=6
OM 2090 FOR S=16 TO 32 STEP 4:SOUND 0,S,1
   4,10:DL=EA*EA*EA:SOUND 0,0,0,0:EA=1^0:
   NEXT S:GOTO 2110
NH 2110 RL=7*(L=1)+9*(L=2)+15*(L=3)+15*(L
   =4)+12*(L=5):POSITION 7,10:? L:POSITIO
   N 11,12:? "0 "
JG 2120 GOSUB 1870:GOTO 730
UC 2130 FOR W=A TO A1 STEP ST:POSITION W,
   4:? B$(19):NEXT W
RS 2140 FOR W=A TO A1 STEP ST:POSITION W+
   V,4:? B$(19):POSITION W,4:? G$(19):NEX
   T W:GOTO 730
ZJ 2150 FOR W=A TO A1 STEP ST:POSITION W,
   4:? B$:NEXT W
WI 2160 FOR W=A TO A1 STEP ST:POSITION W+
   V,4:? B$:POSITION W,4:? A$:NEXT W:GOTO
    730
UO 2170 FOR W=A TO A1 STEP ST:POSITION W,
   4:? B$(19):NEXT W
WI 2180 FOR W=A TO A1 STEP ST:POSITION W+
   V,4:? B$(19):POSITION W,4:? F$:NEXT W:
   GOTO 730
ZV 2190 FOR W=A TO A1 STEP ST:POSITION W,
   4:? B$:NEXT W
KG 2200 FOR W=A TO A1 STEP ST:POSITION W+
   V,4:? B$:POSITION W,4:? G$:NEXT W:GOTO
    730
IE 2210 FOR W=A TO A1 STEP ST:POSITION W,
   7:? B$(10):NEXT W
UT 2220 FOR W=A TO A1 STEP ST:POSITION W+
   V,R:? B$(10):POSITION W,R:? F$:NEXT W:
   GOTO 730
VP 2230 FOR W=A TO A1 STEP ST:POSITION W,
   10:? B$(19):NEXT W
GH 2240 FOR W=A TO A1 STEP ST:POSITION W+
   V,10:? B$(19):POSITION W,10:? G$(19):N
   EXT W:GOTO 730
XB 2250 POSITION A,4:? G$
QS 2260 A1=22*(ST=1)+16*(ST=-1)
DN 2270 FOR W=A TO A1 STEP ST:POSITION W,
   4:? #5
IU 2280 POSITION W+ST,4:? G$:NEXT W:POSIT
   ION A,4:? G$:IF N=3 THEN OP=0:DM=0:GOT
   O 1210
VB 2290 GOTO 1690
WO 2300 DATA 0,3,87,82,87,82,82,69,81,68,
   65,69,90,3,69,124,124,10,124,124,81,82
   ,88,3,90,88,67,3,68,81,82,69,81,87
LZ 2310 DATA 82,69,90,67,81,67,90,83,82,8
   8,82,87,88,69,3,124,81,3,87,3,69,124,9
   0,88,88,82,88,82,88,67
RT 2320 DATA 81,87,3,81,3,87,87,69,65,68,
   124,124,65,68,3,124,124,6,124,65,3,90,
   83,67,3,67,65,68,65,87,67,3,90,82
VC 2330 DATA 67,90,3,124,81,68,81,87,82,8
   7,88,67,124,10,124,3,82,3,69,81,68,124
   ,90,88,82,88,88,67,90,67,
GS 2340 DATA 81,3,87,3,87,87,87,3,10,65,8
   3,87,67,3,67,124,65,83,3,88,69,124,81,
   3,3,67,124,81,3,68,65,69,8,81,88,68
XR 2350 DATA 90,83,83,68,3,83,87,3,69,3,6
   7,6,8,124,90,83,88,67,81,68,3,88,3,88,
   3,82,67,3
WS 2360 DATA 81,3,69,3,81,3,87,69,6,90,83
   ,83,88,67,65,3,90,3,67,65,82,3,90,69,8
   1,69,81,3,69,90,87,3,3,68,65,67,124
SQ 2370 DATA 81,88,68,65,67,3,87,3,88,87,
   3,10,81,67,124,65,69,65,68,90,3,82,3,6
   7,90,3,67
TH 2380 DATA 81,87,82,82,82,82,69,3,90,3,
   81,3,69,3,90,67,81,67,124,81,83,88,82,
   3,3,81,68,3,124,3,82,67,90,88,67,65
UD 2390 DATA 67,124,81,3,81,69,8,90,69,65
   ,88,68,6,65,3,81,3,65,3,124,90,67,90,8
```

continued on next page

```
   8,82,88,82,67
RP 2400 IF STRIG(0)<>0 THEN 2400
TB 2410 POKE 704,PEEK(710):POKE 53277,0:R
   UN
```

education

# MUSIC FLASHCARDS

Article on page 66.

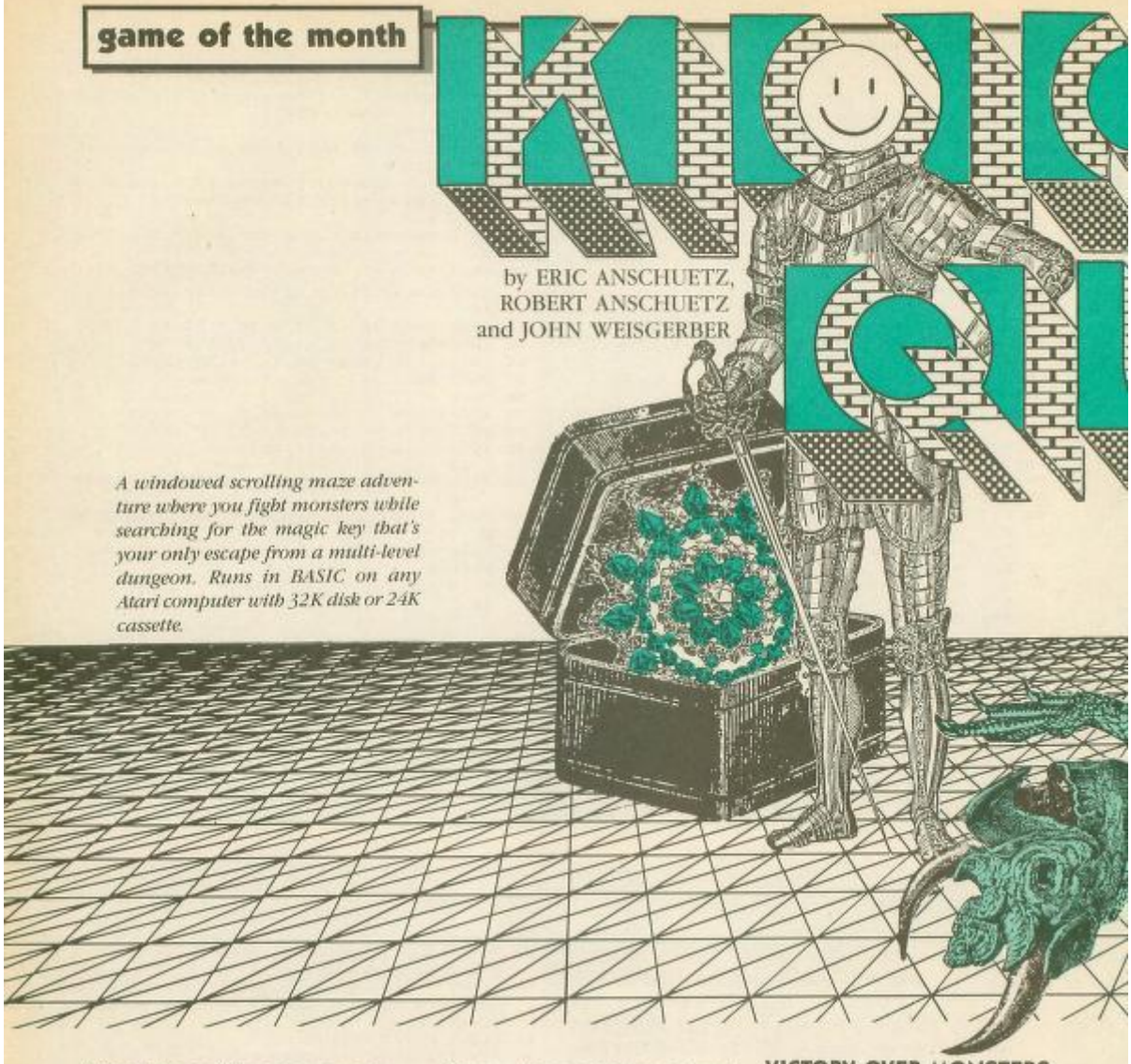
```
SF 10 REM MUSIC FLASHCARDS
UM 20 REM BY JAMES W. BROWN
RN 30 REM ANTIC MAGAZINE
IC 100 DIM INOTE(23),CNOTE$(23),KCODE(23)
ZX 110 DIM N$(1),RNG$(1)
NK 115 REM TITLE SCREEN
SY 120 GRAPHICS 1
MW 121 SETCOLOR 0,4,6:SETCOLOR 1,8,6:SETC
   OLOR 2,0,12:SETCOLOR 3,12,6
EG 123 POSITION 0,1:? #6;"■■■■■■■■■■■■■■
   ■■■■"
LR 124 POSITION 0,2:? #6;"■
       ■"
ST 125 POSITION 0,3:? #6;"■ MU■■IC F■■Sh■■
   R■■ ■"
CU 127 POSITION 0,4:? #6;"■
       ■"
OB 130 POSITION 0,5:? #6;"■          ■V
       ■"
NZ 132 POSITION 0,6:? #6;"■
       ■"
TU 135 POSITION 0,7:? #6;"■  Ja■■S W. Br■
   ■N  ■"
FI 137 POSITION 0,8:? #6;"■
       ■"
JL 140 POSITION 0,9:? #6;"■         18■■
       ■"
DW 141 REM ASTER,7 SPACES,CTRL-Y,INV 8,IN
   V-CTRL-S,7 SPACES,INV-CTRL-J
HO 142 POSITION 0,10:? #6;"■
        ■"
LD 144 POSITION 0,11:? #6;"■
        ■"
TN 145 POSITION 0,12:? #6;"■■■■■■■■■■■■■
   ■■■■"
EI 200 REM SET UP NOTE VALUES
UZ 205 RESTORE 1000
TD 210 FOR I=1 TO 23
XL 215 READ N:INOTE(I)=N:NEXT I
TF 220 FOR I=1 TO 23
CA 225 READ N$:CNOTE$(I,I)=N$:NEXT I
TH 230 FOR I=1 TO 23
JU 235 READ N:KCODE(I)=N:NEXT I
EA 236 FOR J=23 TO 1 STEP -1:SOUND 0,INOT
   E(J),10,8:FOR D=1 TO 50:NEXT D:GOSUB 5
   00:NEXT J
OJ 238 FOR J=1 TO 21:SOUND 0,INOTE(J),10,
   8:FOR D=1 TO 50:NEXT D:GOSUB 500:NEXT
   J
MC 240 CH=764:REM KEYCODE BUFFER
VK 245 CRSINH=752:SHFLOK=702:OLDSCOR=-1
GT 249 REM START MAIN LOOP
AD 250 DEL=50:MAXTIM=100:SCORE=0
RC 255 POKE CRSINH,1
XN 260 ? "How many notes do you want";:GO
   SUB 1500
CO 265 TRAP 1265:INPUT NRNDS:TRAP 40000
NE 267 IF NRNDS<1 THEN 260
VY 269 GRAPHICS 0:POKE SHFLOK,64
IG 270 ? :? "Enter ■■ for treble (G clef
   ) only."
PC 275 ? :? "Enter ■■ for bass (F clef)
   only."
OE 277 ? :? "Enter ■■ for whole staff."
SG 280 GOSUB 1500:TRAP 1280:INPUT RNG$:TR
   AP 40000
WA 285 IF RNG$<>"S" AND RNG$<>"F" AND RNG
   $<>"G" THEN 270
EF 299 REM DRAW STAFF
ZE 300 GRAPHICS 7:COLOR 1
FA 301 SETCOLOR 0,10,8:REM STAFF
ZP 302 SETCOLOR 1,7,10:REM NOTE (AND TEXT
    LUMINANCE)
FV 303 SETCOLOR 2,8,2:REM TEXT WINDOW
WM 304 SETCOLOR 4,8,2:REM BACKGROUND
PO 305 FOR CLEF=1 TO 2
WO 310 FOR LINE=1 TO 5
BB 315 Y=6*LINE+36*CLEF-32
RN 320 PLOT 1,Y:DRAWTO 159,Y
FA 325 NEXT LINE
ZK 327 NEXT CLEF
JJ 330 RESTORE 1050:READ X,Y:PLOT X,Y
UN 340 FOR POINT=1 TO 14
BC 345 READ X,Y:DRAWTO X,Y
EE 350 NEXT POINT
KE 360 RESTORE 1060:READ X,Y:PLOT X,Y
VG 365 FOR POINT=1 TO 14
AT 370 READ X,Y:DRAWTO X,Y
EX 375 NEXT POINT
ZL 380 PLOT 28,49:PLOT 28,55
RJ 385 POKE CRSINH,1
HI 400 FOR ROUND=1 TO NRNDS
HN 402 REM PICK A NOTE AND SHOW IT
TF 405 N=1+INT(RND(0)*23):SOUND 0,0,0,0
CO 406 IF RNG$="F" AND N<14 THEN 405
EB 407 IF RNG$="G" AND N>13 THEN 405
EX 410 GOSUB 900:REM DRAW IT
JT 412 WRONG=0
MO 413 REM WAIT FOR KEYPRESS
AE 415 ITIM=MAXTIM
NM 420 ITIM=ITIM-1:FOR D=1 TO DEL:NEXT D
CW 422 IF ITIM<1 THEN 440
ZA 425 ? "■";ITIM,"NOTE? ":GOSUB 1500
KV 430 IF PEEK(CH)=255 THEN 420
TN 435 IF PEEK(CH)=KCODE(N) THEN 460
ZY 437 REM WRONG - BUZZ
MA 439 WRONG=WRONG+1
EI 440 SOUND 0,85,12,8:POKE CH,255
WS 445 FOR D=1 TO DEL:NEXT D:SOUND 0,0,0,
   0
AE 446 IF WRONG>2 THEN ITIM=0
JL 447 IF ITIM<1 THEN 465
NT 450 GOTO 420
PX 459 REM RIGHT - PLAY THE NOTE
XF 460 SOUND 0,INOTE(N),10,8
XX 461 NN=INOTE(N):IF NN=0 THEN NN=254
XS 462 SOUND 0,NN,10,8
SJ 464 REM RIGHT OR TIMEOUT -- SHOW IT
UR 465 ? "■";ITIM,"NOTE IS ";CNOTE$(N,N)
ZZ 470 FOR D=1 TO DEL*4:NEXT D:POKE CH,25
   5
VZ 475 GOSUB 800:SCORE=SCORE+ITIM
```

game of the month

# KOOKY'S QUEST

by ERIC ANSCHUETZ,
ROBERT ANSCHUETZ
and JOHN WEISGERBER

*A windowed scrolling maze adventure where you fight monsters while searching for the magic key that's your only escape from a multi-level dungeon. Runs in BASIC on any Atari computer with 32K disk or 24K cassette.*



Trapped in an evil dungeon, your only hope of escaping is to descend to the lowest level, retrieve the magic key, and use it to unlock the portal at the uppermost level.

Check the program with TYPO II as you type it in. SAVE the game before you RUN it. Playing Kooky's Quest uses the keyboard and a joystick in port one. There are two skill levels. In the beginner game, you're aided by a displayed map of the dungeon, which is erased for the expert game.

To move from one dungeon level to another, you must obtain enough treasures—rings and chests. You need only one treasure to descend from the first level to the second, but you'll need five to reach each of the deepest dungeons.

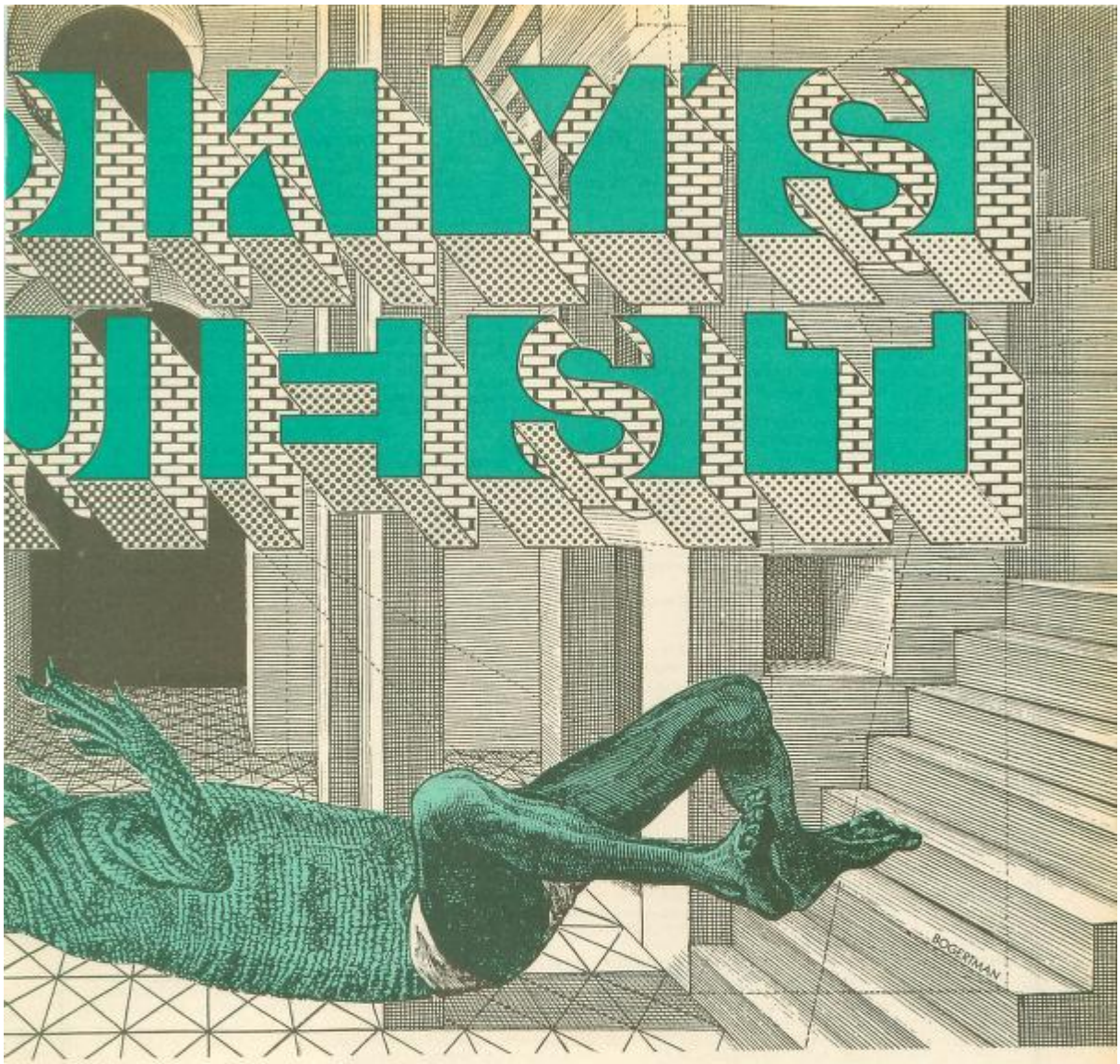
Once you have enough treasures to change levels, go to the up or down staircase, shown with a D or U on the map. If you are playing without the map, you can identify a staircase by sound effects, because the stairs aren't in your view.

## VICTORY OVER MONSTERS

Getting treasures is not an easy task. In each room, you will find a treasure and a monster. It is important to get to the treasure before the monster gets it, before you kill the monster or the monster kills you. In each case, you'll lose the treasure.

If you clear all the rooms on any level and don't have enough treasure, you'll be forced to wander that level forever—or until you stop the game.

You have ten lives for the entire game. If you'd like more, rejuvenate




yourself at line 610.

You'll be asked to choose a weapon as you enter a room. Available weapons are listed at the bottom of the screen and can be selected with numbers. The axe, mace and sword are swinging weapons—press the fire button and move the joystick from left or right.

The knife, lightning bolt and fireball are throwing weapons. They are used in the same way as swinging weapons—except they continue moving in a straight line until they hit something.

Choose your weapons with care. Each has different properties. You'll find that some require six hits to kill, others only one hit.

Note that while fighting, you can only attack to the left or right, and can't protect yourself from above or below.

You can encounter six types of monsters—vampire bats, giant hobgoblins, wicked skeletons, venomous serpents, decaying mummies and malevolent spiders that inhabit only the lowest level. Your adventurer, Kooky is shown as a happy face symbol.

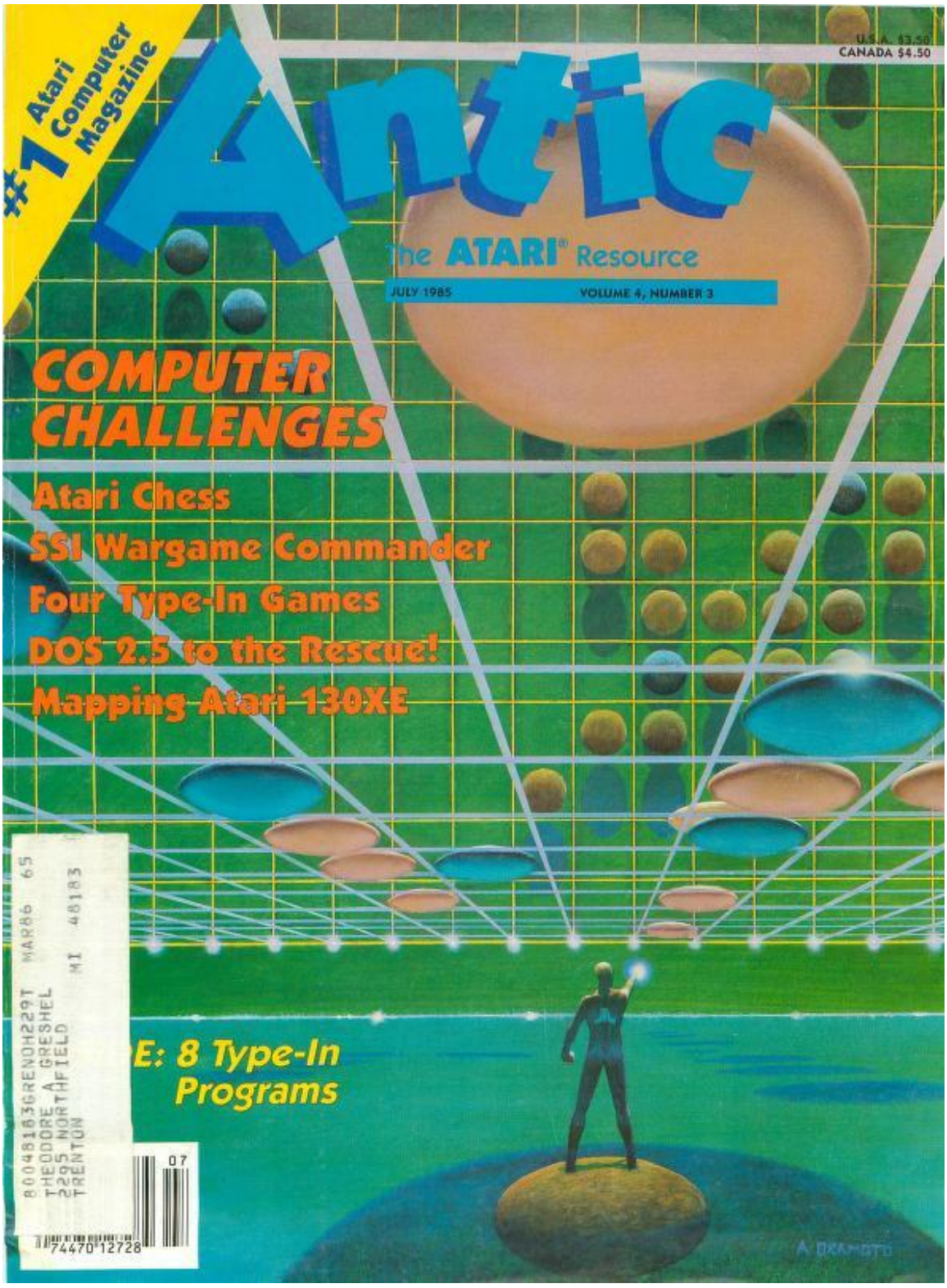
Instead of escaping the dungeon, you may choose to accumulate points. Treasures are worth five points, defeated monsters ten.

*All three authors are 19-year-old computer science majors at Eastern Michigan University.*

Listing on page 51.

# OVERFLOW

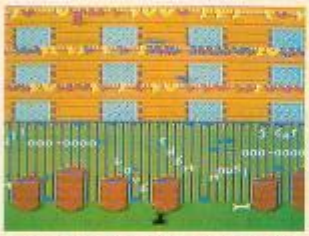
**ANTIC VOL. 4, NO. 3 / JULY 1985**



Antic®
The ATARI® Resource
JULY 1985, VOLUME 4, NUMBER 3



The economics of wargames . . . . . 28



New games from Synapse . . . . . 16



Guess that song. Or else . . . . . 24



Build golf courses in your spare time! . . 46

## FEATURES

**ANTIC ANNIVERSARY ALBUM** 10
Atari world comes to Antic's party

**EIGHT NEW SYNAPSE GAMES** by Jack Powell 16
Synapse merges with Broderbund

**130XE: HOW ATARI'S NEW 8-BIT DOES IT** by Jack Powell 18
XL compatibility plus 128K power

**CRYPTOQUOTES** by Michael Fleischmann 21
"Automatic blackboard" for code breaking TYPE-IN SOFTWARE

**GUESS THAT SONG** by Karl Wiegers 24
How many notes do **YOU** bid? TYPE-IN SOFTWARE

**WAR GAMES** by Jack Powell 28
The story of Strategic Simulations Inc.

**SLIDE** by Mark Moore and Michael Ciraolo 33
Can **YOU** solve the 24-puzzle? TYPE-IN SOFTWARE

**COMPUTER CHESS PROGRAMMING** by Michael Ciraolo 35
How good is your Atari's game?

**EVERYTHING YOU WANTED TO KNOW ABOUT EVERY DOS** by Eric Clausen 40
Including brand-new Atari DOS 2.5

## DEPARTMENTS

EDUCATION
**DUCKS, SEAHORSES AND MUPPETS** by Anita Malnig
New educational software 13

BONUS GAME
**STAR VENTURE** by Ronald Saari
Easy type-in game with fast action TYPE-IN SOFTWARE 23

BONUS GAME
**DARKSTAR** by Michael Mitchell
Zapping aliens with radioactive waste TYPE-IN SOFTWARE 31

BONUS GAME
**OVERFLOW** by Anschuetz, Anschuetz & Weisgerber
Only your plumber's helper can save you TYPE-IN SOFTWARE 39

GAME OF THE MONTH
**MINIATURE GOLF** by David Plotkin
Plus golf course construction set TYPE-IN SOFTWARE 46

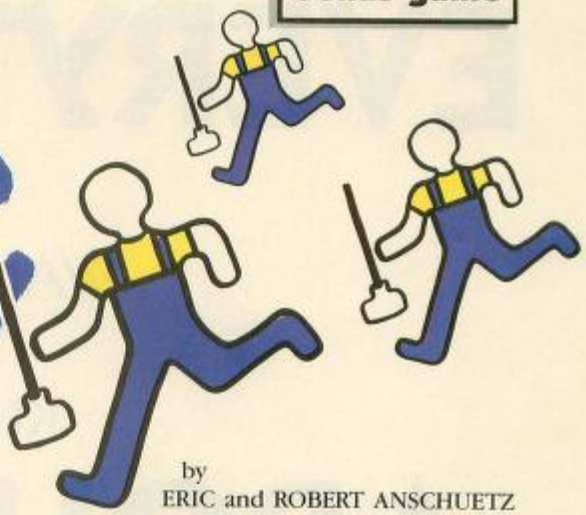
## SOFTWARE LIBRARY

**TYPE-IN LISTINGS SECTION** 51

I/O BOARD 6
HELP 8
PRODUCT REVIEWS 74
ATARI SERVICE CENTERS 78
SHOPPERS' GUIDE 81
ADVERTISERS' LIST 81
NEW PRODUCTS 82
ANTIC ARCADE CATALOG AC1

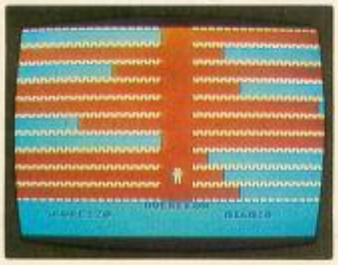
bonus game

# OVER FLOW



*Only your plumber's helper can save you!*

by
ERIC and ROBERT ANSCHUETZ
and JOHN WEISGERBER





*Discover the thrills and excitement of being a Sanitary Engineer (plumber) as you enter a maze of killer pipes. This BASIC program works on all Atari computers of any memory size, with disk or cassette. Just type in Listing 1, check it with TYPO II and SAVE a copy before you RUN.*

Dear Miss Goodmanners:

Is there a correct way to stem a rising tide of bathroom back-ups while entertaining distinguished guests in my 49-room mansion?

Concerned

Dear Concerned:

Miss Goodmanners prefers the term water closet . . . Sadly, overflow problems are part of our modern age and must be dealt with firmly and quickly. Miss Goodmanners would never commit the faux pas of embarassing guests during a sumptuous banquet by confronting them with a mass of raw sewage.

Miss Goodmanners would award you credit for decisively hurling your plumber's helper at each of the overflowing water closets, while tactfully running between pipelines without scaring the guests. Of course, sewage problems tend to get worse as a party wears on—especially if guests discover that the duck pate has disagreed with them.

With apologies to our favorite etiquette columnist, this program uses players for the plumber, plunger and central pipe. All vertical motion is produced using a string printed at the correct place in memory. The main graphics mode is ANTIC 5.

An unusual technique here is the use of basically the same memory address for every line of the playfield graphics. Each line has its own LMS and the high byte on every one is set to the same page. The water is moved back and forth by incrementing the low byte on each LMS. The increment for each line is held in a one-dimensional array. If a column of sewage is hit by the plunger, the increment's sign is simply reversed.

*Members of the noted* ***Antic*** *game programming team of Anschuetz, Anschuetz, and Weisgerber tell us that they are currently attending Eastern Michigan University and would like jobs as programmers when they graduate.*

Listing on page 62

```
   TION 5,5:? #6;"SCORE ";SC:X=X+1
ST 270 SETCOLOR 0,X,X:IF X>350 THEN GOSUB
    300:GOTO 40
AO 280 IF STRIG(0)=0 THEN GOTO 80
MY 290 SH=2:GOTO 260
VT 300 FOR D=0 TO 3:SOUND D,0,0,0:NEXT D:
   RETURN
```

bonus game

# OVERFLOW

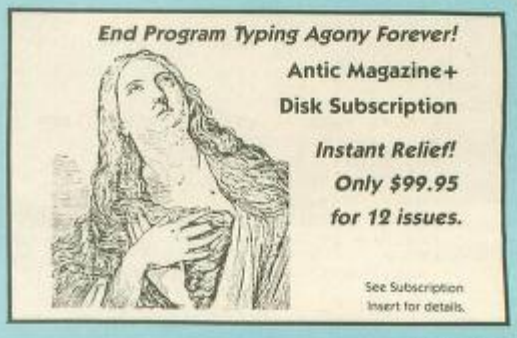
Article on page 39.

## LISTING 1

Don't type the TYPO II Codes!

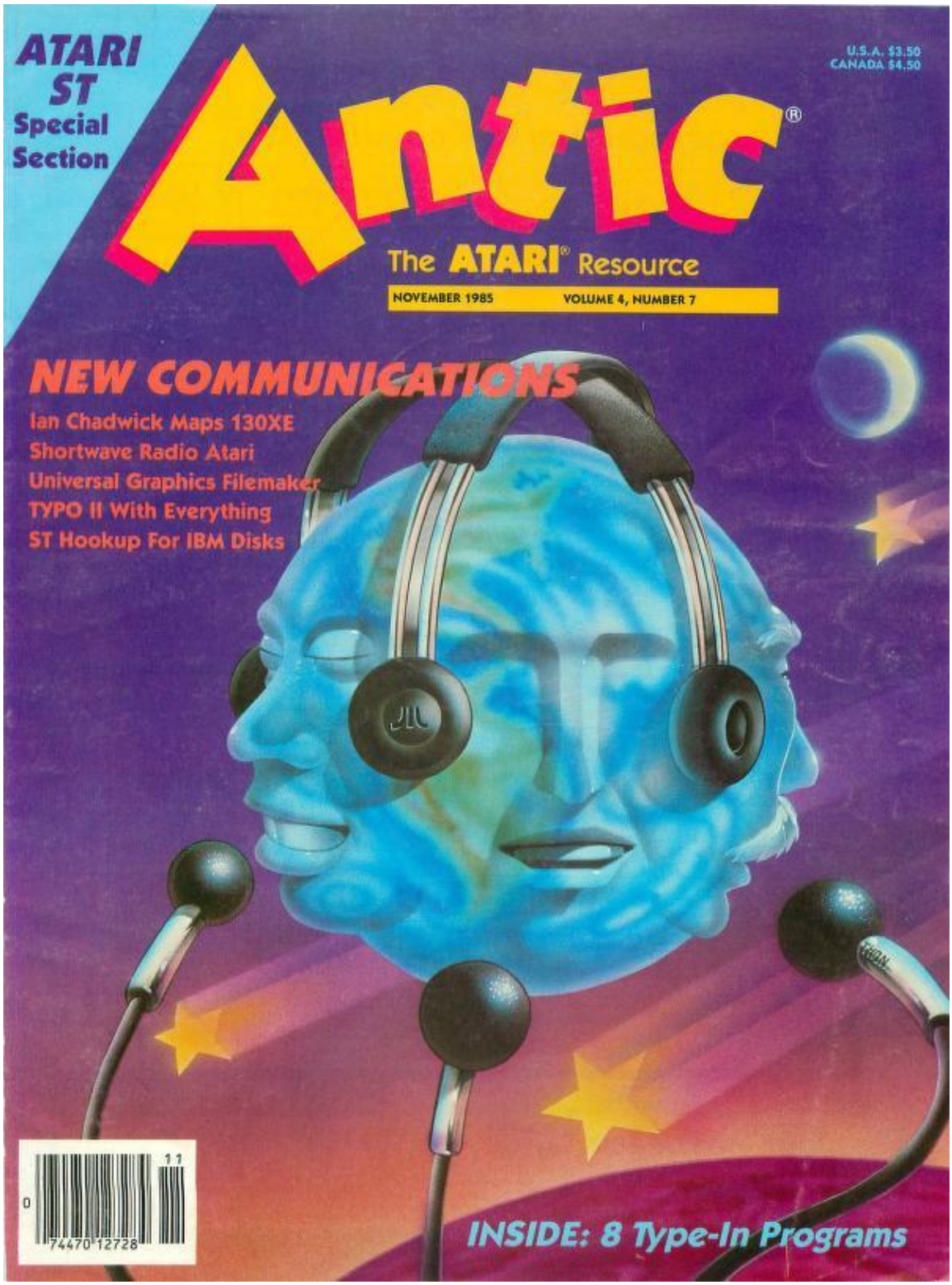
```
BX 10 REM OVERFLOW
FU 20 REM BY ANSCHUETZ/WEISGERBER/ANSCHUE
   TZ
FW 30 REM (c) 1985, ANTIC PUBLISHING
RC 40 TUNL=58:WATR=148:MT=0
CO 50 POKE 106,PEEK(740)
HP 60 MEM=PEEK(106):DL=(MEM-5)*256:CH=(ME
   M-4)*256:JW=MEM-12:EA=MEM-11:RA=(MEM-9
   )*256:S=(MEM-7)*256
MA 70 GRAPHICS 2:POKE 709,58:DIM O(10),AS
   (46),B$(16):POKE 755,0:POSITION 6,4:?
   #6;"OVERFLOW"
DU 80 POKE 710,38:POKE 712,38:? :? "    A
   nschuetz/Weisgerber/Anschuetz":GOTO 41
   0
NM 90 POKE 709,TUNL:POKE 710,WATR:POKE 71
   2,0:POKE 707,TUNL
DW 100 POKE 89,JW:POKE 756,MEM-4:POKE 559
   ,62:POKE 560,0:POKE 561,MEM-5:POKE 532
   51,128:POKE 53248,123
RN 110 ? :? :? :? :FOR X=1 TO 10:O(X)=INT
   (RND(0)*23)*2-1:NEXT X:SC=0:POKE 656,0:
   POKE 657,15:? "OVERFLOW"
OM 120 POKE 657,4:? "SCORE:0     ":POKE 6
   57,26:? "HIGH:";HC;:YX=33:POKE 88,YX:P
   OSITION 0,0:? #6;A$(17,38)
KN 130 OL=DL+1:C=0:FOR X=DL+4 TO DL+31 ST
   EP 3:Y=INT(RND(0)*11)+101:Y=Y+(Y/2<>IN
   T(Y/2)):POKE X,Y:NEXT X
SF 140 C=C+1:OL=OL+3:POKE OL,PEEK(OL)+O(C
   )
UZ 150 POKE 53278,0:IF OL>DL+28 THEN OL=D
   L+1:C=0
FJ 160 ST=STICK(0):IF ST=14 THEN YX=YX-16
   *(YX>33):POKE 88,YX:POSITION 0,0:? #6;
   A$(17,46)
VO 170 POKE 709,TUNL:POKE 710,WATR
LT 180 IF ST=13 THEN YX=YX+16*(YX<177):PO
   KE 88,YX-16:POSITION 0,0:? #6;A$(1,38)

CY 190 IF ST=7 AND J=0 THEN PLX=133:PLY=Y
   X+3:J=4:POKE 89,EA:POKE 88,YX+3:POSITI
   ON 0,0:? #6;B$(1,8):POKE 89,JW
DU 200 IF ST=11 AND J=0 THEN PLX=115:PLY=
   YX+3:J=-4:POKE 89,EA:POKE 88,YX+3:POSI
   TION 0,0:? #6;B$(9,16):POKE 89,JW
NW 210 IF PEEK(53255)>3 THEN POKE 707,WAT
   R:GOTO 360
OO 220 IF J<>0 THEN 240
NF 230 GOTO 140
CD 240 IF PEEK(53253)>3 THEN 270
WO 250 POKE 53249,PLX:PLX=PLX+J
DT 260 IF PEEK(53253)<4 THEN 290
VT 270 HO=(PLY-20)/16:IF SGN(O(HO))<>SGN(
   J) THEN 330
NH 280 SOUND 0,100,10,10:SC=SC+10:O(HO)=-
   O(HO):POKE 657,10:? SC;:GOTO 310
BC 290 IF PLX>200 OR PLX<48 THEN 330
NA 300 GOTO 140
KY 310 POKE 77,0:IF SC=250 THEN SOUND 0,6
   0,10,12:GOTO 350
FN 320 IF SC=500 THEN SOUND 0,60,10,12:PO
   KE 53259,3:POKE 53251,112:FOR X=1 TO 1
   00:NEXT X
UU 330 POKE 53249,0:J=0:POKE 89,EA:POKE 8
   8,PLY:POSITION 0,0:? #6;A$(1,8):POKE 8
   9,JW:POKE 53278,0
MW 340 SOUND 0,0,0,0:GOTO 140
SP 350 FOR X=1 TO 10:O(X)=O(X)*2:NEXT X:F
   OR X=DL+4 TO DL+31 STEP 3:Y=PEEK(X):PO
   KE X,Y+(Y/2=INT(Y/2)):NEXT X:GOTO 330
DO 360 J=0:POKE 656,3:POKE 657,9:? "PRESS
    <START> TO BEGIN":: IF SC>HC THEN HC=S
   C
MP 370 YC=YC+(YC<255)-255*(YC=255):POKE 7
   84,YC:IF PEEK(53279)<>6 THEN 370
FY 380 FOR X=DL+4 TO DL+31 STEP 3:POKE X,
   188:NEXT X:POKE 89,EA:POKE 88,PLY:POSI
   TION 0,0:? #6;A$(1,8):POKE 53251,128
YD 390 POKE 764,198
LT 400 POKE 89,JW:POKE 88,YX:POSITION 0,0
   :? #6;"                ":POKE 707,TUNL:
   POKE 53249,0:POKE 53259,1:GOTO 110
TM 410 FOR X=0 TO 511:POKE X+CH,PEEK(X+57
   344):NEXT X:FOR X=0 TO 23:READ A:POKE
   X+CH,A:NEXT X
SX 420 FOR X=DL TO DL+2:POKE X,112:NEXT X
   :FOR X=DL+3 TO DL+30 STEP 3:POKE X,69:
   POKE X+1,108:POKE X+2,5/256:NEXT X
EE 430 POKE DL+33,66:POKE DL+34,PEEK(660)
   :POKE DL+35,PEEK(661):FOR X=DL+36 TO D
   L+38:POKE X,2:NEXT X:POKE DL+39,65
OO 440 FOR X=RA+32 TO RA+191:POKE X,255:N
   EXT X:FOR X=JW*256 TO RA:POKE X,0:NEXT
    X:FOR X=1 TO 46:A$(X,X)=" ":NEXT X
HV 450 FOR X=17 TO 30:READ A:A$(X)=CHR$(A
   +32):NEXT X:A$(46,46)=" ":FOR X=1 TO 1
   6:READ A:B$(X)=CHR$(A):NEXT X
VF 460 POKE 623,1:POKE 54279,MEM-16:POKE
   53277,3:POKE 704,198:POKE 705,198:POKE
    707,TUNL:POKE 53259,1
NU 470 FOR X=5 TO 5+100:POKE X,2:POKE X+1
   47,2:NEXT X:FOR X=5+109 TO 5+146:POKE
   X,1:NEXT X:SOUND 1,160,8,2:GOTO 90
KY 480 DATA 84,170,170,170,170,170,170,69
   ,84,255,255,255,255,255,255,69
XV 490 DATA 14,14,14,4,31,31,14,14,14,14,
   10,10,10,10,33,35,35,255,255,35,35,33,
   160,128,128,255,255,128,128,160
```

# ROBOT DUNGEON

**ANTIC VOL. 4, NO. 7 / NOVEMBER 1985 (Disk Subscription Only)**



continued from page 76

Final Legacy offers six levels of play. The higher a level, the more enemy ships and missile sites you will encounter. The different graphics screens are quite inventive and the sound warnings are effective. The 8-page manual is very good.

In some ways this game reminds me of a more advanced **Missile Command**. The new game's main problem is that it lacks strategy and tends to become a little repetitious. But overall, Final Legacy is quite entertaining. It demands alertness and quick responses, and is a worthy successor to the great tradition of Atari action games.

## CHOPPER HUNT

Imagic
981 University Avenue
Los Gatos, CA 95030
(800) 654-7340
32K disk
$19.95

*Reviewed by Jack Powell*

I don't want to take much space reviewing this throwback, but **Antic** has a certain obligation to warn readers against junk products. **Chopper Hunt** would make a very nice public domain game and might have been seriously considered as a professional program when the Atari 800 was just a twinkle in Nolan Bushnell's eye.

You move a slow, blocky helicopter back and forth across a crude, blocky screen, bombing holes in the ground so you can get to the flashing grey squares. Meanwhile, a slow, blocky enemy airplane flies back and forth dropping slow, blocky "dirt balls". I had to look twice to make sure my BASIC cartridge was not in the machine slowing down the action.

If you plunk down your hard-earned money for this outdated arcade game, you are helping prove that Barnum was right to say, "There's a sucker born every minute." The circus has come to town and you are in the center ring wearing funny hair.

## REFORGER '88

Strategic Simulations, Inc.
883 Stierlin Rd, Bldg. A-200
Mountain View, CA 94043
(415) 964-1200
$59.95, 48K disk

*Reviewed by Dr. John F. Stanoch*

Ever since the end of WWII, there has been constant tension along the border between East and West Germany. It is here that the two superpowers have deployed their most advanced war technology. SSI's new **Reforger '88** is an operational-level game of NATO defense against a combined Soviet and East German thrust through the Fulda Gap into the American-defended sector of West Germany.

To win, the Soviet player must gain as many victory points as possible, while the NATO player must hold down the number of points given up. Points are awarded to both players for enemy unit destruction and possession of West German cities. However, once a NATO city has been overrun, the NATO player can not regain victory points for recapturing it. The Soviet player receives a substantial bonus for capturing the main U.S. Airbase at Frankfurt. The NATO point total is subtracted from the Soviet total to arrive at the final score.

In the solitaire mode, the computer ably controls Soviet forces. Units are combined into combat groups that can be combined or split apart at the start of a turn. Orders are given and then all movements and combat are simultaneously executed.

To win, airpower must be successfully integrated with ground troops. Adequate anti-aircraft defense is also necessary. Enemy combat groups are displayed only if adjacent to a friendly combat group or sighted by friendly air recon missions. Many times, my NATO forces were unpleasantly surprised to find a few Soviet armor combat groups racing toward Frankfurt, well behind what seemed an adequate defense line!

I have one valuable suggestion. Before you play, go out and buy a good set of fine-point overhead projection markers. These will prove indispensible for marking the position of all units on the SSI plasticized data/map card. It is next to impossible to competently play Reforger '88 without an overview map in front of you. Many times, since the screen displays only a fraction of the entire scrolling map, a player may forget what one combat group was ordered to do on one section of the map.

Although Reforger '88 is a complicated game, its excitement is well worth the effort. I recommend this game to any advanced wargamer. But keep a cold drink available, you will probably work up a sweat playing it.

Antic
The ATARI Resource
DISK BONUS
November 1985
ROBOT DUNGEON
A Deadly 1,200-room Maze
Only one way out . . .
but it changes every time!

# A/W/A BASIC CODE LISTINGS



# ANSCHUETZ HIGH SCORE / GAME LISTER

```
10 GRAPHICS 0:SETCOLOR 2,0,0:POKE 82,0
:POKE 53774,64:POKE 16,64:POKE 752,1
15 DIM D$(1),GAME$(25),A$(25),NAME$(25
),ARCADE$(25),B$(25),X$(25),AA$(25000)
,TEMP$(25),NEWHIGH$(7),SC$(7)
16 DIM DATA$(40),HISCORE$(7),FAKE$(7),
HOLD$(7),INI$(3),DEV$(14),I$(3)
20 ? "{CLEAR}  ANSCHUETZ HIGH SCORE/GAME L
ISTER":? :? :? :? "(1) RETRIEVE HIGH S
CORE":? "(2) VIEW GAMES"
30 ? "(3) ADD NEW GAME":? "(4) EDIT HI
GH SCORE":? "(5) VIEW TREASURE CHESTS"
:? "(6) DOS":? :?
40 TRAP 90:? :INPUT C:IF C=1 THEN TRAP
 40000:GOTO 100
45 IF C=2 THEN TRAP 40000:GOTO 400
50 IF C=3 THEN TRAP 40000:GOTO 200
60 IF C=4 THEN TRAP 40000:GOTO 300
70 IF C=5 THEN TRAP 40000:GOTO 700
80 IF C=6 THEN TRAP 40000:DOS
90 TRAP 40000:GOTO 20
100 ? "{CLEAR}":? "NAME OF GAME";:INPUT GAME
$
102 IF LEN(GAME$)=25 THEN 105
103 FOR Q=LEN(GAME$)+1 TO 25:GAME$(Q,Q
)=" ":NEXT Q
105 CLOSE #1:OPEN #1,4,0,"D:GAMSCORE.D
AT"
110 TRAP 150:INPUT #1;A$:IF A$=GAME$ T
HEN 170
120 GOTO 110
150 TRAP 40000:? :? GAME$:? "DOESN'T E
XIST.":? "CHECK LISTING.":GOTO 190
170 INPUT #1;HISCORE$:INPUT #1;INI$:?
:? "THE HIGH SCORE OF":? GAME$:? "IS "
;HISCORE$;" BY ";INI$
190 CLOSE #1:? :? :? "       PRESS {RETURN} FOR MAIN MENU"
191 POKE 764,255
192 IF PEEK(764)=12 THEN POKE 764,255:
GOTO 20
193 GOTO 192
200 ? "{CLEAR}":? "ENTER NAME OF NEW GAME:";
? "(UP TO 25 CHARACTERS)":? :INPUT NAM
E$:CLOSE #1
203 IF LEN(NAME$)=25 THEN 210
204 IF LEN(NAME$)>25 THEN 200
205 FOR Q=LEN(NAME$)+1 TO 25:NAME$(Q,Q
)=" ":NEXT Q
210 OPEN #1,9,0,"D:GAMSCORE.DAT":? #1;
NAME$:? #1;"0000000":? #1;"AAA"
220 CLOSE #1:? :? :? "       PRESS {RETURN} FOR MAIN MENU"
225 POKE 764,255
226 IF PEEK(764)=12 THEN POKE 764,255:
GOTO 20
227 GOTO 226
300 ? "{CLEAR}":? "ENTER NAME OF GAME":? "(U
P TO 25 CHARACTERS LONG)":? :INPUT ARC
ADE$
301 ? :? "NEW HIGH SCORE (UP TO 7 DIGI
TS)":? :INPUT NEWHIGH$:? :? "INITIALS
(UP TO 3 LETTERS)":? :INPUT I$
302 IF LEN(NEWHIGH$)=0 OR LEN(ARCADE$)
=0 OR LEN(I$)=0 THEN 300
303 IF LEN(NEWHIGH$)>7 OR LEN(ARCADE$)
>25 OR LEN(I$)>3 THEN 300
304 IF LEN(NEWHIGH$)=7 THEN 310
305 NH=LEN(NEWHIGH$):HOLD$=NEWHIGH$:FO
R M=NH TO 6:LET NEWHIGH$(M-NH+1)="0":N
EXT M
306 LET NEWHIGH$(LEN(NEWHIGH$)+1)=HOLD
$
310 IF LEN(ARCADE$)=25 THEN 315
311 FOR Q=LEN(ARCADE$)+1 TO 25:ARCADE$
(Q,Q)=" ":NEXT Q
315 IF LEN(I$)=3 THEN 319
316 FOR Q=LEN(I$)+1 TO 3:I$(Q,Q)=" ":N
EXT Q
319 CLOSE #1:OPEN #1,12,0,"D:GAMSCORE.
DAT":TRAP 360
320 INPUT #1;B$:IF B$=ARCADE$ THEN 380

330 GOTO 320
360 TRAP 40000:? :? :? ARCADE$:? "DOES
N'T EXIT":GOTO 390
380 ? #1;NEWHIGH$:? #1;I$
390 CLOSE #1:? :? :? "       PRESS {RETURN} FOR MAIN MENU"
391 POKE 764,255
392 IF PEEK(764)=12 THEN POKE 764,255:
GOTO 20
393 GOTO 392
400 ? "{CLEAR}":? "{A}LPHABETICAL OR":? "{S}EQUE
NTIAL WITH HIGH SCORES OR":? "{M}AKE ALP
HABETICAL LIST":? :INPUT D$
410 IF D$<>"A" AND D$<>"S" AND D$<>"M"
 THEN 400
420 IF D$="M" THEN 500
425 IF D$="A" THEN 800
430 ? "{CLEAR}":CLOSE #1:OPEN #1,4,0,"D:GAMS
CORE.DAT"
440 POKE 201,1:TRAP 450:INPUT #1;X$:?
X$,:INPUT #1;SC$:? SC$,:INPUT #1;INI$:
? INI$:GOTO 440
450 POKE 201,10:TRAP 40000:? :? "THAT'
S ALL FOLKS!!!"
460 CLOSE #1:? :? :? "       PRESS {RETURN} FOR MAIN MENU":? "   OR PRESS {R} TO
 VIEW THE LIST AGAIN"
470 POKE 764,255
471 IF PEEK(764)=12 THEN POKE 764,255:
GOTO 20
472 IF PEEK(764)=40 THEN POKE 764,255:
GOTO 430
473 GOTO 471
500 ? "{CLEAR}":SLEN=25:SLEN1=24:AA$="  ":AA$
(25000)=" ":AA$(2)=AA$:OPEN #1,4,0,"D:
GAMSCORE.DAT"
505 ? "  PLEASE WAIT WHILE I READ IN G
AMES..."
510 I=1:FOR J=1 TO 1000:TRAP 550:INPUT
 #1;TEMP$
520 AA$(I,I+SLEN1)=TEMP$:I=I+SLEN:INPU
T #1;FAKE$:INPUT #1;FAKE$:NEXT J
550 TRAP 40000:ENTRIES=J-1
555 CLOSE #1
560 GOSUB 1000
565 TRAP 40000:? "{CLEAR}":? "ENTER DEVISE T
O LIST TO";:INPUT DEV$:IF DEV$(1,1)="E
" OR DEV$(1,1)="S" THEN CLOSE #1
566 IF DEV$(1,1)="E" OR DEV$(1,1)="S"
THEN OPEN #1,8,0,DEV$:SETCOLOR 2,0,0:P
OKE 53774,64:POKE 16,64
567 IF DEV$(1,1)="E" OR DEV$(1,1)="S"
THEN POKE 752,1*DEV$(1,1)="E":GOTO 570

568 TRAP 565:IF DEV$(1,1)="D" THEN CLO
SE #1:OPEN #1,8,0,"D:OURGAMES.DAT":GOT
O 570
569 TRAP 565:CLOSE #1:OPEN #1,8,0,DEV$

570 TRAP 40000:I=1:FOR K=1 TO ENTRIES:
? #1;"(";K;") ";AA$(I,I+SLEN1):I=I+SLE
N:NEXT K:? #1;
580 ? #1;"THAT'S ALL FOLKS!!!":IF DEV$
(1,1)="S" THEN POKE 752,1
590 CLOSE #1:? :? :? "       PRESS {RETURN} FOR MAIN MENU":? "   OR PRESS {R} TO
 VIEW THE LIST AGAIN"
593 POKE 764,255
594 IF PEEK(764)=12 THEN POKE 764,255:
GOTO 20
595 IF PEEK(764)=40 THEN POKE 764,255:
GOTO 565
596 GOTO 594
700 ? "{CLEAR}":CLOSE #1:OPEN #1,4,0,"D:TCHE
ST.DAT"
730 TRAP 750:INPUT #1;DATA$:? DATA$:GO
TO 730
750 ? :? "THAT'S ALL FOLKS!!!"
755 CLOSE #1:? :? :? "       PRESS {RETURN} FOR MAIN MENU":? "   OR PRESS {R} TO
 VIEW THE LIST AGAIN"
756 POKE 764,255
757 IF PEEK(764)=12 THEN POKE 764,255:
GOTO 20
758 IF PEEK(764)=40 THEN POKE 764,255:
GOTO 700
759 GOTO 757
800 ? "{CLEAR}":CLOSE #1:OPEN #1,4,0,"D:OURG
AMES.DAT"
830 TRAP 855:INPUT #1;DATA$:? DATA$:GO
TO 830
855 CLOSE #1:? :? :? "       PRESS {RETURN} FOR MAIN MENU":? "   OR PRESS {R} TO
 VIEW THE LIST AGAIN"
856 POKE 764,255
857 IF PEEK(764)=12 THEN POKE 764,255:
GOTO 20
858 IF PEEK(764)=40 THEN POKE 764,255:
GOTO 800
859 GOTO 857
1000 RESTORE 1010:FOR A5=1568 TO 1693:
READ A6:POKE A5,A6:NEXT A5
1010 DATA 104,104,133,217,104,133,216,
104,133,209,104,133,208,169,0,133,218,
133,207,162,1,165,216,133
1020 DATA 214,165,217,133,215,24,165,2
14,133,212,101,205,133,214,165,215,133
,213,105,0,133,215,164,203
1030 DATA 165,206,240,10,177,214,209,2
12,144,44,240,12,176,19,177,214,209,21
2,144,13,240,2,176,30
1040 DATA 200,196,204,240,227,176,23,1
44,223,169,1,133,218,164,205,136,177,2
14,72,177,212,145,214,104
1050 DATA 145,212,192,0,208,241,232,22
4,0,208,2,230,207,228,208,208,172,165,
209,197,207,208,166,165
1060 DATA 218,201,0,208,144,96
1100 ? :? :? " NOW WAIT WHILE I ALPHAB
ETIZE THE LIST"
1105 POKE 203,0:POKE 204,24:POKE 205,2
5:POKE 206,0:PP=USR(1568,ADR(AA$),ENTR
IES)
1110 RETURN
```

# ALIEN ASSAULT

```
100 GRAPHICS 1:SETCOLOR 2,0,0:POKE 16,
64:POKE 53774,64:DL=PEEK(560)+256*PEEK
(561):POKE DL+14,23:POKE DL+18,2
110 POKE 54276,4:SETCOLOR 0,8,4:POSITI
ON 5,9:? #6;"ALIEN ASSAULT":POSITION 4
,11:? #6;"press start to begin"
120 POSITION 9,13:? #6;"ANSCHUETZ/WEIS
GERBER/ANSCHUETZ"
130 IF PEEK(53279)<>6 THEN 130
140 GRAPHICS 1:POKE 559,0:POKE 16,64:P
OKE 53774,64:PM=12288:OP=10:SETCOLOR 0
,14,2:SETCOLOR 1,13,10:SETCOLOR 2,12,6

150 POKE 755,0:FOR A=12800 TO 12834:RE
AD D:POKE A,D:NEXT A:DATA 112
160 DATA 112,112,112,112,112,112,112,1
12,112,112,112,112,69,0,58,66,35,51,69
,0,59,66,35,51,2,2,2,2,2,66,0,0,2,2,2
170 POKE 560,0:POKE 561,50:POKE 12830,
PEEK(660):POKE 12831,PEEK(661)
180 GOTO 450
190 ML=INT(RND(0)*2)+1:MX=INT(RND(0)*7
84)+120:MY=10:OS=0:NM=NM-1:IF NM<0 THE
N POKE 77,0:GOTO 410
200 IF MX<(PE+20)*4 THEN POKE 656,0:PO
KE 657,18:? "WEST"
210 IF MX>(PE+20)*4 THEN POKE 656,0:PO
KE 657,18:? "EAST"
220 POKE PLY+ML,MY:MY=MY+2+(MIAD*0.25)
:IF MY>130 THEN POKE PLX+ML,0:GOTO 350

230 IF STICK(0)=7 THEN PE=PE+2*(PE<216
)
240 IF STICK(0)=11 THEN PE=PE-2*(PE>0)

250 IF STICK(0)=14 THEN CHY=CHY-(CHY>4
0)*4:POKE PLY,CHY
260 IF STICK(0)=13 THEN CHY=CHY+(CHY<1
28)*4:POKE PLY,CHY
270 IF STRIG(0)=0 THEN POKE 53278,0:PO
KE PLY+3,CHY+1:POKE PLX+3,124:EA=EA-1:
POKE 656,1:POKE 657,7:GOTO 300
280 POKE 12814,PE/2:POKE 12820,PE
290 SMX=MX-PE*4+40+OS:OS=OS+(ML=1)*-2+
(ML=2)*2:POKE PLX+ML,(SMX>48)*(SMX<202
)*SMX+(MIAD*0.25):GOTO 220
300 ? EA;" ":SOUND 0,100,10,8:SOUND 0,
102,10,8:SOUND 0,105,10,8:SOUND 0,110,
10,8:SOUND 0,0,0,0
310 IF PEEK(53260) THEN SOUND 0,200,2,
10:FOR X=0 TO 20:NEXT X:SOUND 0,0,0,0:
POKE PLX+ML,0:SC=SC+100
320 POKE PLX+3,0:POKE PLY+3,250:POKE 6
56,1:POKE 657,18:? SC;:IF PEEK(53260)
THEN 190
330 IF EA<1 THEN 760
340 GOTO 280
350 HP=INT((MX+OS)/4-0.5)+256*59:IF PE
EK(HP)=13 THEN POKE HP,26:SOUND 0,255,
10,10:FOR X=1 TO 25:NEXT X
360 IF PEEK(HP)=26 THEN SOUND 0,0,0,0:
GOTO 190
370 IF PEEK(HP)=11 THEN POKE HP,14:POK
E HP+1,15:MSL=MSL+1:GOTO 390
380 IF PEEK(HP)=12 THEN POKE HP,15:POK
E HP-1,14:MSL=MSL+1
390 IF MSL>OP THEN 760
400 POKE 656,1:POKE 657,31:? MSL;:SOUN
D 0,150,4,10:FOR X=1 TO 50:NEXT X:SOUN
D 0,0,0,0:GOTO 190
410 FOR LN=1 TO 3:SOUND LN,0,0,0:NEXT
LN:GH=GH+1:MIAD=MIAD+1:PE=108:NM=MIAD*
5+10:OP=OP-0.5:IF NM>40 THEN NM=40
420 IF GH>15 THEN GH=0
430 SC=SC+(NS-MSL)*10:POKE 656,1:POKE
657,18:? SC;:EA=50:MSL=0:FOR JW=1 TO 3
00:NEXT JW:POKE 559,0
440 POKE 656,1:POKE 657,7:? "50":POKE
656,1:POKE 657,31:? "0":GOTO 590
450 RESTORE 630:FOR I=1536 TO 1706:REA
D A:POKE I,A:NEXT I:FOR I=1774 TO 1787
:POKE I,0:NEXT I
460 FOR I=PM+1023 TO PM+2047:POKE I,0:
NEXT I:FOR I=PM+1025 TO PM+1031:READ A
:POKE I,A:NEXT I
470 FOR I=PM+1281 TO PM+1296:READ A:PO
KE I,A:NEXT I:FOR I=PM+1793 TO PM+1797
:READ A:POKE I,A:NEXT I
480 FOR I=PM+1537 TO PM+1552:READ A:PO
KE I,A:NEXT I:FOR I=PM+1799 TO PM+1992
:POKE I,8:NEXT I
490 POKE 704,132:POKE 705,70:POKE 706,
70:POKE 707,12:PLX=53248:PLY=1780:PLL=
1784:POKE 623,1
500 POKE 1788,PM/256+4:POKE 53277,2:PO
KE 54279,PM/256:Q=USR(1696):POKE PLL,7
:POKE PLL+1,16:POKE PLL+2,16
510 POKE PLL+3,200:POKE 53257,2:POKE 5
3258,2:POKE 53259,2
520 POKE PLX,124:POKE PLY,80:CHY=80:PO
KE PLY+3,221
530 PE=108:EA=50:NM=10:CH=PM:FOR I=0 T
O 511:POKE CH+I,PEEK(57344+I):NEXT I
540 FOR X=8 TO 127:READ A:POKE CH+X,A:
NEXT X:FOR X=208 TO 215:READ A:POKE CH
+X,A:NEXT X:POKE 756,CH/256:X=14848
550 POKE X,INT(RND(0)*2)+1:X=X+1:PN=IN
T(RND(0)*5)+1:FOR X=X TO X+PN:POKE X,I
NT(RND(0)*6)+5:NEXT X
560 POKE X,INT(RND(0)*2)+3:IF X<14995
THEN X=X+1:GOTO 550
570 POKE 656,1:POKE 657,7:? "50
   0          0"
580 POKE 656,2:POKE 657,5:? "ENERGY
    SCORE     DESTROYED"
590 FOR X=15104 TO 15359:IF INT(RND(0)
*5)+1=1 THEN POKE X,11:X=X+1:POKE X,12
:NS=NS+1:NEXT X:GOTO 610
600 POKE X,13:NEXT X
610 POKE 12814,54:POKE 12820,108:POKE
559,62:SOUND 1,199,4,3:SOUND 2,200,1,4
:SOUND 3,201,2,3
620 SETCOLOR 4,GH*2,8*(GH<>0):GOTO 190

630 DATA 162,3,189,244,6,240,89,56,221
,240,6,240,83,141,254,6,106,141,255,6,
142,253,6,24,169,0,109,253,6,24,109
640 DATA 252,6,133,204,133,206,189,240
,6,133,203,173,254,6,133,205,189,248,6
,170,232,46,255,6,144,16,168,177
650 DATA 203,145,205,169,0,145,203,136
,202,208,244,76,87,6,160,0,177,203,145
,205,169,0,145,203,200,202,208,244
660 DATA 174,253,6,173,254,6,157,240,6
,189,236,6,240,48,133,203,24,138,141,2
53,6,109,235,6,133,204,24,173,253
670 DATA 6,109,252,6,133,206,189,240,6
,133,205,189,248,6,170,160,0,177,203,1
45,205,200,202,208,248,174,253,6,169
680 DATA 0,157,236,6,202,48,3,76,2,6,7
6,98,228,0,0,104,169,7,162,6,160,0,32,
92,228,96
690 DATA 28,8,65,99,65,8,28,24,24,24,2
4,28,28,63,63,63,63,120,120,224,224,19
2,192,34,20,8,28,34
700 DATA 24,24,24,24,24,56,56,252,252,252
,252,30,30,7,7,3,3
710 DATA 0,0,0,0,1,5,23,127,0,0,1,5,5,
61,63,255,0,0,0,0,64,80,208,255,0,0,0,
64,80,112,245,255
720 DATA 0,2,5,85,87,127,255,255,160,1
68,88,85,85,223,255,255,0,128,164,85,9
3,127,255,255
730 DATA 0,0,160,149,85,85,253,255,32,
164,84,85,87,127,255,255,2,10,137,85,2
23,255,255,255
740 DATA 255,245,212,80,84,213,251,239
,255,95,87,23,95,111,187,254,255,255,2
55,255,255,255,255,255,255
750 DATA 255,255,255,255,209,68,209,24
5,255,255,255,127,31,71,31,255,255,255
,255,255,115,69,81,247
760 FOR RA=0 TO 3:POKE PLX+RA,0:SOUND
RA,0,0,0:NEXT RA:GRAPHICS 17:POKE 16,6
4:POKE 53774,64
770 POSITION 5,10:? #6;"game over":POS
ITION 4,13:? #6;"score: ";SC:GOTO 770
```

# ALIEN ASSAULT - 2017

```
50 REM ERIC ANSCHUETZ - FIXED HARD COD
ED MEMORY LOCATIONS THAT CONFLICTED WI
TH DOS AND PREVENTED RUNNING 7/19/17
60 REM EEA ANYWHERE IN THE CODE THAT R
EFERENCES THE VARIABLE "BASE" IS NEW C
ODE THAT REPLACED SPECIFIC MEM LOCS
100 GRAPHICS 1:SETCOLOR 2,0,0:POKE 16,
64:POKE 53774,64:DL=PEEK(560)+256*PEEK
(561):POKE DL+14,23:POKE DL+18,2
110 POKE 54276,4:SETCOLOR 0,8,4:POSITI
ON 5,9:? #6;"ALIEN ASSAULT":POSITION 4
,11:? #6;"press start to begin"
120 POSITION 9,13:? #6;"ANSCHUETZ/WEIS
GERBER/ANSCHUETZ"
130 IF PEEK(53279)<>6 THEN 130
132 GRAPHICS 1:POKE 106,PEEK(106)-16:R
EM EEA MODIFY RAMTOP TO SAFELY STORE P
M, CHAR SET, AND DISPLAY LIST
135 BASE=PEEK(106):PM=BASE*256:REM STO
RE GRAPHICS AT TOP, NOT IN THE MIDDLE
OF DOS!
140 GRAPHICS 1:POKE 559,0:POKE 16,64:P
OKE 53774,64:OP=10:SETCOLOR 0,14,2:SET
COLOR 1,13,10:SETCOLOR 2,12,6
150 POKE 755,0:FOR A=(BASE+2)*256 TO (
BASE+2)*256+34:READ D:POKE A,D:NEXT A:
DATA 112
160 DATA 112,112,112,112,112,112,112,1
12,112,112,112,112,69,0,0,66,35,0,69,0
,0,66,35,0,2,2,2,2,66,0,0,2,2,2
162 REM EEA DISPLAY LIST ABOVE USED TO
 HAVE HARDCODED MEMORY LOCATIONS TO JU
MP TO.  MODIFY MSB AFTER DL LOADED
165 POKE (BASE+2)*256+15,BASE+10:POKE
(BASE+2)*256+18,BASE+3:POKE (BASE+2)*2
56+21,BASE+11
167 POKE (BASE+2)*256+24,BASE+3
170 POKE 560,0:POKE 561,BASE+2:POKE (B
ASE+2)*256+30,PEEK(660):POKE (BASE+2)*
256+31,PEEK(661)
180 GOTO 450
190 ML=INT(RND(0)*2)+1:MX=INT(RND(0)*7
84)+120:MY=10:OS=0:NM=NM-1:IF NM<0 THE
N POKE 77,0:GOTO 410
200 IF MX<(PE+20)*4 THEN POKE 656,0:PO
KE 657,18:? "WEST"
210 IF MX>(PE+20)*4 THEN POKE 656,0:PO
KE 657,18:? "EAST"
220 POKE PLY+ML,MY:MY=MY+2+(MIAD*0.25)
:IF MY>130 THEN POKE PLX+ML,0:GOTO 350

230 IF STICK(0)=7 THEN PE=PE+2*(PE<216
)
240 IF STICK(0)=11 THEN PE=PE-2*(PE>0)

250 IF STICK(0)=14 THEN CHY=CHY-(CHY>4
0)*4:POKE PLY,CHY
260 IF STICK(0)=13 THEN CHY=CHY+(CHY<1
28)*4:POKE PLY,CHY
270 IF STRIG(0)=0 THEN POKE 53278,0:PO
KE PLY+3,CHY+1:POKE PLX+3,124:EA=EA-1:
POKE 656,1:POKE 657,7:GOTO 300
280 POKE (BASE+2)*256+14,PE/2:POKE (BA
SE+2)*256+20,PE
290 SMX=MX-PE*4+40+OS:OS=OS+(ML=1)*-2+
(ML=2)*2:POKE PLX+ML,(SMX>48)*(SMX<202
)*SMX+(MIAD*0.25):GOTO 220
300 ? EA;" ":SOUND 0,100,10,8:SOUND 0,
102,10,8:SOUND 0,105,10,8:SOUND 0,110,
10,8:SOUND 0,0,0,0
310 IF PEEK(53260) THEN SOUND 0,200,2,
10:FOR X=0 TO 20:NEXT X:SOUND 0,0,0,0:
POKE PLX+ML,0:SC=SC+100
320 POKE PLX+3,0:POKE PLY+3,250:POKE 6
56,1:POKE 657,18:? SC;:IF PEEK(53260)
THEN 190
330 IF EA<1 THEN 760
340 GOTO 280
350 HP=INT((MX+OS)/4-0.5)+256*(BASE+11
):IF PEEK(HP)=13 THEN POKE HP,26:SOUND
 0,255,10,10:FOR X=1 TO 25:NEXT X
360 IF PEEK(HP)=26 THEN SOUND 0,0,0,0:
GOTO 190
370 IF PEEK(HP)=11 THEN POKE HP,14:POK
E HP+1,15:MSL=MSL+1:GOTO 390
380 IF PEEK(HP)=12 THEN POKE HP,15:POK
E HP-1,14:MSL=MSL+1
390 IF MSL>OP THEN 760
400 POKE 656,1:POKE 657,31:? MSL;:SOUN
D 0,150,4,10:FOR X=1 TO 50:NEXT X:SOUN
D 0,0,0,0:GOTO 190
410 FOR LN=1 TO 3:SOUND LN,0,0,0:NEXT
LN:GH=GH+1:MIAD=MIAD+1:PE=108:NM=MIAD*
5+10:OP=OP-0.5:IF NM>40 THEN NM=40
420 IF GH>15 THEN GH=0
430 SC=SC+(NS-MSL)*10:POKE 656,1:POKE
657,18:? SC;:EA=50:MSL=0:FOR JW=1 TO 3
00:NEXT JW:POKE 559,0
440 POKE 656,1:POKE 657,7:? "50":POKE
656,1:POKE 657,31:? "0":GOTO 590
450 RESTORE 630:FOR I=1536 TO 1706:REA
D A:POKE I,A:NEXT I:FOR I=1774 TO 1787
:POKE I,0:NEXT I
460 FOR I=PM+1023 TO PM+2047:POKE I,0:
NEXT I:FOR I=PM+1025 TO PM+1031:READ A
:POKE I,A:NEXT I
470 FOR I=PM+1281 TO PM+1296:READ A:PO
KE I,A:NEXT I:FOR I=PM+1793 TO PM+1797
:READ A:POKE I,A:NEXT I
480 FOR I=PM+1537 TO PM+1552:READ A:PO
KE I,A:NEXT I:FOR I=PM+1799 TO PM+1992
:POKE I,8:NEXT I
490 POKE 704,132:POKE 705,70:POKE 706,
70:POKE 707,12:PLX=53248:PLY=1780:PLL=
1784:POKE 623,1
500 POKE 1788,PM/256+4:POKE 53277,2:PO
KE 54279,PM/256:Q=USR(1696):POKE PLL,7
:POKE PLL+1,16:POKE PLL+2,16
510 POKE PLL+3,200:POKE 53257,2:POKE 5
3258,2:POKE 53259,2
520 POKE PLX,124:POKE PLY,80:CHY=80:PO
KE PLY+3,221
530 PE=108:EA=50:NM=10:CH=PM:FOR I=0 T
O 511:POKE CH+I,PEEK(57344+I):NEXT I
540 FOR X=8 TO 127:READ A:POKE CH+X,A:
NEXT X:FOR X=208 TO 215:READ A:POKE CH
+X,A:NEXT X:POKE 756,CH/256
545 X=(BASE+10)*256
550 POKE X,INT(RND(0)*2)+1:X=X+1:PN=IN
T(RND(0)*5)+1:FOR X=X TO X+PN:POKE X,I
NT(RND(0)*6)+5:NEXT X
560 POKE X,INT(RND(0)*2)+3:IF X<((BASE
+10)*256+147) THEN X=X+1:GOTO 550
570 POKE 656,1:POKE 657,7:? "50
  0          0"
580 POKE 656,2:POKE 657,5:? "ENERGY
   SCORE    DESTROYED"
590 FOR X=(BASE+11)*256 TO (BASE+11)*2
56+255
595 IF INT(RND(0)*5)+1=1 THEN POKE X,1
1:X=X+1:POKE X,12:NS=NS+1:NEXT X:GOTO
610
600 POKE X,13:NEXT X
610 POKE (BASE+2)*256+14,54:POKE (BASE
+2)*256+20,108:POKE 559,62
615 SOUND 1,199,4,3:SOUND 2,200,1,4:SO
UND 3,201,2,3
620 SETCOLOR 4,GH*2,8*(GH<>0):GOTO 190

630 DATA 162,3,189,244,6,240,89,56,221
,240,6,240,83,141,254,6,106,141,255,6,
142,253,6,24,169,0,109,253,6,24,109
640 DATA 252,6,133,204,133,206,189,240
,6,133,203,173,254,6,133,205,189,248,6
,170,232,46,255,6,144,16,168,177
650 DATA 203,145,205,169,0,145,203,136
,202,208,244,76,87,6,160,0,177,203,145
,205,169,0,145,203,200,202,208,244
660 DATA 174,253,6,173,254,6,157,240,6
,189,236,6,240,48,133,203,24,138,141,2
53,6,109,235,6,133,204,24,173,253
670 DATA 6,109,252,6,133,206,189,240,6
,133,205,189,248,6,170,160,0,177,203,1
45,205,200,202,208,248,174,253,6,169
680 DATA 0,157,236,6,202,48,3,76,2,6,7
6,98,228,0,0,104,169,7,162,6,160,0,32,
92,228,96
690 DATA 28,8,65,99,65,8,28,24,24,24,2
4,28,28,63,63,63,63,120,120,224,224,19
2,192,34,20,8,28,34
700 DATA 24,24,24,24,56,56,252,252,252
,252,30,30,7,7,3,3
710 DATA 0,0,0,0,1,5,23,127,0,0,1,5,5,
61,63,255,0,0,0,0,64,80,208,255,0,0,0,
64,80,112,245,255
720 DATA 0,2,5,85,87,127,255,255,160,1
68,88,85,85,223,255,255,0,128,164,85,9
3,127,255,255
730 DATA 0,0,160,149,85,85,253,255,32,
164,84,85,87,127,255,255,2,10,137,85,2
23,255,255,255
740 DATA 255,245,212,80,84,213,251,239
,255,95,87,23,95,111,187,254,255,255,2
55,255,255,255,255,255
750 DATA 255,255,255,255,209,68,209,24
5,255,255,255,127,31,71,31,255,255,255
,255,255,115,69,81,247
760 FOR RA=0 TO 3:POKE PLX+RA,0:SOUND
RA,0,0,0:NEXT RA:GRAPHICS 17:POKE 16,6
4:POKE 53774,64
770 POSITION 5,10:? #6;"game over":POS
ITION 4,13:? #6;"score: ";SC:GOTO 770
```

# EMU TRACK MEET DATABASE

```
15 ? "ß":POKE 752,1:SETCOLOR 2,0,0:POK
E 82,0:POKE 16,64:POKE 53774,64
100 POKE 559,0:POSITION 0,0:? "╔═══════
═══════════════════════════════":? "
╠═══════════ MAIN MENU ═══════════╣"
105 FOR Y=2 TO 19:POSITION 0,Y:? "╠":P
OSITION 39,Y:? "╠╣":NEXT Y:? "╚═══════
═══════ a/w/a ═══════════════╝"
110 POSITION 2,4:? "1. START NEW     4
. PARTICIPANT DATA":POSITION 5,5:? "TR
ACK MEET"
115 POSITION 2,8:? "2. EVENT DATA    5
. CREATING HEATS":POSITION 2,11:? "3.
AFFILIATION    6. RESULTS"
117 POSITION 9,12:? "DATA"
120 POSITION 11,16:? "MAKE SELECTION O
R":POSITION 9,17:? "HIT D FOR DESCRIPT
IONS":POKE 559,34
130 CLOSE #1:OPEN #1,4,0,"K:":GET #1,K
:IF K=68 THEN 200
133 TRAP 155
135 IF K=49 THEN CLOSE #1:RUN "D:START
NEW.BAS"
136 IF K=50 THEN CLOSE #1:RUN "D:EVENT
DAT.BAS"
137 IF K=51 THEN CLOSE #1:RUN "D:AFFDA
TA.BAS"
138 IF K=52 THEN CLOSE #1:RUN "D:PARTD
ATA.BAS"
139 IF K=53 THEN CLOSE #1:RUN "D:CREAT
EHT.BAS"
140 IF K=54 THEN CLOSE #1:RUN "D:RESUL
TS.BAS"
150 GOTO 130
155 CLOSE #1:OPEN #1,4,0,"K:"
160 TRAP 40000:? "ß":POSITION 2,8:? "ß
ß         INSERT PROGRAM DISK":? "
       AND HIT RETURN":POKE 764,255
165 IF PEEK(764)<>12 THEN 165
170 ? "ß":GOTO 100
200 TRAP 40000:POKE 559,0:? "ß"
210 GOSUB 300:POSITION 10,3:? "START N
EW TRACK MEET":POSITION 4,5:? "ENTER G
ENERAL DATA FOR NEW MEET"
211 POSITION 6,6:? "REVISE DATA ON EXI
STING MEET":POSITION 12,13:? "PRESS N
FOR NEXT":GOSUB 500:POKE 559,34:GOSUB
400
215 POKE 559,0:? "ß":GOSUB 300:POSITIO
N 15,3:? "EVENT DATA":POSITION 11,5:?
"ENTER/REVISE EVENTS"
216 POSITION 11,6:? "PRINT OR LIST EVE
NT":POSITION 12,13:? "PRESS N FOR NEXT
"
218 GOSUB 500:POKE 559,34:GOSUB 400
220 POKE 559,0:? "ß":GOSUB 300:POSITIO
N 12,3:? "AFFILIATION DATA":POSITION 8
,5:? "ENTER/REVISE AFFILIATIONS"
221 POSITION 10,6:? "DISPLAY AFFILIATI
ONS":POSITION 11,7:? "PRINT AFFILIATIO
NS":POSITION 12,13:? "PRESS N FOR NEXT
":GOSUB 500
222 POKE 559,34:GOSUB 400
225 POKE 559,0:? "ß":GOSUB 300:POSITIO
N 12,3:? "PARTICIPANT DATA":POSITION 1
0,5:? "ENTER/REVISE ENTRANTS"
226 POSITION 11,6:? "LIST/PRINT ENTRAN
TS":POSITION 12,13:? "PRESS N FOR NEXT
":GOSUB 500:POKE 559,34:GOSUB 400
230 POKE 559,0:? "ß":GOSUB 300:POSITIO
N 13,3:? "CREATING HEATS"
231 POSITION 11,5:? "SEEDING OF ENTRAN
TS":POSITION 12,6:? "PRINT HEAT SHEETS
"
232 POSITION 12,13:? "PRESS N FOR NEXT
":GOSUB 500:POKE 559,34:GOSUB 400
235 POKE 559,0:? "ß":GOSUB 300:POSITIO
N 17,3:? "RESULTS":POSITION 10,5:? "EN
TER/REVISE RESULTS"
236 POSITION 14,6:? "PRINT RESULTS":PO
SITION 10,7:? "CROSS COUNTRY RESULTS":
POSITION 12,13
237 ? "PRESS N FOR NEXT":GOSUB 500:POK
E 559,34:GOSUB 400:GOTO 200
300 POSITION 0,0:? "╔═══════════════════
═══════════════════":? "╠
═══ DESCRIPTIONS ═══════════╣"
301 FOR Y=2 TO 19:POSITION 0,Y:? "╠":P
OSITION 39,Y:? "╠╣":NEXT Y:? "╚═══════
═══════ a/w/a ═══════════════╝":RET
URN
400 GET #1,K:IF K<>78 AND K<>77 THEN 4
00
401 IF K=78 THEN RETURN
402 POP :? "ß":GOTO 100
500 POSITION 12,14:? "PRESS M FOR MAIN
":RETURN
```

# EMU TRACK MEET DATABASE – AFFILIATIONS

```
10 DIM A$(25),B$(20),C1$(20),C2$(20),C
3$(20),PH$(15),CN$(20),K$(2),T$(1),G$(
25),G1$(20),G2$(15):FLAG=0
15 DIM D5$(5),SC1$(6),SC2$(6),SC3$(6),
SP$(39),G3$(5),G4$(6):SP$=" ":SP$(39)=
" ":SP$(2)=SP$
20 POSITION 0,0:FOR X=1 TO 23:? "■";:N
EXT X
50 POKE 752,1:? "↰":SETCOLOR 2,0,0:POS
ITION 0,0:POKE 82,0:POKE 16,64:POKE 53
774,64:TRAP 40000
76 POSITION 0,8:? "       INSERT THE D
ATA DISKETTE":? "          HIT RETURN TO
 CONTINUE";:OPEN #1,4,0,"K:":GET #1,X
77 TRAP 80:CLOSE #1:OPEN #2,4,0,"D:TRA
CK.BAS":? "■■↰":CLOSE #2:GOTO 76
80 ? "↰":TRAP 40000:POSITION 0,11:? "H
IT E TO EDIT/VIEW EXISTING AFFILIATION
":? "      HIT N TO ADD NEW AFFILIATION
"
81 ? "   HIT L TO LIST/PRINT AFFILIATI
ONS"
84 TRAP 80:? "                    WHICH ←
←";:INPUT T$:IF T$="E" THEN ? :? "  CO
DE NUMBER OF DESIRED AFFILIATION";:INP
UT M
85 TRAP 40000:IF T$<>"E" AND T$<>"N" A
ND T$<>"L" THEN ? "↑↑":GOTO 84
86 TRAP 80:IF T$="L" THEN CLOSE #1:OPE
N #1,4,0,"D:AFFIL.DAT":? "↰":GOTO 700
87 TRAP 80:IF T$="E" THEN CLOSE #1:OPE
N #1,4,0,"D:AFFIL.DAT":CNO=0:FOR X=1 T
O M:GOSUB 500:NEXT X:CLOSE #1:GOTO 99
89 TRAP 91:CLOSE #1:OPEN #1,4,0,"D:AFF
IL.DAT"
90 TRAP 92:CNO=0:FOR X=1 TO 1000:GOSUB
 500:NEXT X
91 TRAP 40000:CLOSE #1:OPEN #1,8,0,"D:
AFFIL.DAT":FLAG=1:CLOSE #1:GOTO 89
92 TRAP 40000:CLOSE #1:CNO=CNO+1:A$="":
B$="":C1$="":C2$="":C3$="":PH$="":CN$
="":D5$="":SC1$="":SC2$="":SC3$=""
99 ? "↰":TRAP 40000:? "          AFFIL
IATION #";CNO;:? :? :POKE 201,3
100 ? " 1. NAME",A$:? " 2. ACRONYM",,B
$:? " 3. ADDRESS L.1",C1$:? " 4. ADDRE
SS L.2",C2$:? " 5. ADDRESS L.3",C3$
105 ? " 6. ZIP CODE",,D5$:? " 7. PHONE
 #",,PH$:? " 8. CONTACT NAME",CN$:? "
9. SCORE:",,,SC1$:? "10. SCORE:",,,SC2
$
110 ? "11. SCORE:",,,SC3$
125 TRAP 99:? :? "          HIT A TO ENTE
R ALL ITEMS":? "   ENTER NUMBER TO EDI
T SINGLE ITEM"
130 ? "       HIT S TO SAVE DATA ON DIS
K":? "       HIT RETURN FOR MAIN MENU"
:? "                 WHICH";:INPUT K$:?
 "↰"
135 IF K$="" THEN CLOSE #1:GOTO 1000
137 IF K$="A" THEN ALL=1:GOTO 140
138 ALL=0
140 IF K$="1" OR ALL THEN ? :? "ENTER
AFFILIATION NAME":INPUT A$
150 IF K$="2" OR ALL THEN ? :? "ENTER
AFFILIATION ACRONYM":INPUT B$
160 IF K$="3" OR ALL THEN ? :? "ENTER
1ST LINE OF ADDRESS":INPUT C1$
170 IF K$="4" OR ALL THEN ? :? "ENTER
2ND LINE OF ADDRESS":INPUT C2$
180 IF K$="5" OR ALL THEN ? :? "ENTER
3RD LINE OF ADDRESS":INPUT C3$
190 IF K$="6" OR ALL THEN ? :? "ENTER
ZIP CODE":INPUT D5$
200 IF K$="7" OR ALL THEN ? :? "ENTER
PHONE NUMBER":INPUT PH$
202 IF K$="8" OR ALL THEN ? :? "ENTER
CONTACT'S NAME":INPUT CN$
205 IF K$="9" OR ALL THEN ? :? "ENTER
SCORE 1":INPUT SC1$
210 IF K$="10" OR ALL THEN ? :? "ENTER
 SCORE 2":INPUT SC2$
215 IF K$="11" OR ALL THEN ? :? "ENTER
 SCORE 3":INPUT SC3$
225 IF K$="S" THEN 300
250 ? "↰":GOTO 99
300 ? " ARE YOU SURE THE ENTRIES ARE C
ORRECT";:INPUT K$:IF K$<>"Y" AND K$<>"
YES" THEN 99
301 IF T$="N" AND FLAG THEN CLOSE #1:O
PEN #1,8,0,"D:AFFIL.DAT":FLAG=0:GOTO 3
06
302 IF T$="N" AND  NOT FLAG THEN CLOSE
 #1:OPEN #1,9,0,"D:AFFIL.DAT":GOTO 306

303 IF T$="E" AND M=1 THEN CLOSE #1:OP
EN #1,12,0,"D:AFFIL.DAT":GOTO 306
305 CLOSE #1:OPEN #1,12,0,"D:AFFIL.DAT
":FOR X=1 TO M-1:GOSUB 520:NEXT X
306 IF LEN(A$)=25 THEN 308
307 FOR Y=LEN(A$)+1 TO 25:A$(Y,Y)=" ":
NEXT Y
308 IF LEN(B$)=20 THEN 310
309 FOR Y=LEN(B$)+1 TO 20:B$(Y,Y)=" ":
NEXT Y
310 IF LEN(C1$)=20 THEN 312
311 FOR Y=LEN(C1$)+1 TO 20:C1$(Y,Y)="
":NEXT Y
312 IF LEN(C2$)=20 THEN 314
313 FOR Y=LEN(C2$)+1 TO 20:C2$(Y,Y)="
":NEXT Y
314 IF LEN(C3$)=20 THEN 316
315 FOR Y=LEN(C3$)+1 TO 20:C3$(Y,Y)="
":NEXT Y
316 IF LEN(PH$)=15 THEN 318
317 FOR Y=LEN(PH$)+1 TO 15:PH$(Y,Y)="
":NEXT Y
318 IF LEN(CN$)=20 THEN 320
319 FOR Y=LEN(CN$)+1 TO 20:CN$(Y,Y)="
":NEXT Y
320 IF LEN(D5$)=5 THEN 322
321 FOR Y=LEN(D5$)+1 TO 5:D5$(Y,Y)=" "
:NEXT Y
322 IF LEN(SC1$)=6 THEN 324
323 FOR Y=LEN(SC1$)+1 TO 6:SC1$(Y,Y)="
 ":NEXT Y
324 IF LEN(SC2$)=6 THEN 326
325 FOR Y=LEN(SC2$)+1 TO 6:SC2$(Y,Y)="
 ":NEXT Y
326 IF LEN(SC3$)=6 THEN 330
327 FOR Y=LEN(SC3$)+1 TO 6:SC3$(Y,Y)="
 ":NEXT Y
330 PRINT #1;A$:PRINT #1;B$:PRINT #1;C
1$:PRINT #1;C2$:PRINT #1;C3$:PRINT #1;
D5$:PRINT #1;PH$:PRINT #1;CN$
331 PRINT #1;SC1$:PRINT #1;SC2$:PRINT
#1;SC3$:CLOSE #1:GOTO 80
500 INPUT #1;A$:INPUT #1;B$:INPUT #1;C
1$:INPUT #1;C2$:INPUT #1;C3$:INPUT #1;
D5$:INPUT #1;PH$:INPUT #1;CN$
515 INPUT #1;SC1$:INPUT #1;SC2$:INPUT
#1;SC3$:CNO=CNO+1:RETURN
520 INPUT #1;G$:INPUT #1;G1$:INPUT #1;
G1$:INPUT #1;G1$:INPUT #1;G1$:INPUT #1
;G3$:INPUT #1;G2$:INPUT #1;G1$
525 INPUT #1;G4$:INPUT #1;G4$:INPUT #1
;G4$:RETURN
700 CLOSE #1:OPEN #1,4,0,"D:AFFIL.DAT"

701 ? "↰":CNO=0:? "       AFFILIATIONS
    AVE. SCORE":? "
_____"
705 POSITION 0,13:? "   _____
_____":? :? "      HIT D T
O DISPLAY AFFILIATIONS"
710 ? "  HIT P TO PRINT LIST OF AFFILI
ATIONS";:? "HIT A TO PRINT ALL INFO. ON
 AFFILIATIONS";
715 ? "         HIT RETURN FOR MAIN MENU
":? "                WHICH";:INPUT K$
720 IF K$="" THEN CLOSE #1:GOTO 1000
725 IF K$="P" THEN 2000
726 IF K$="A" THEN 3000
730 IF K$<>"D" THEN ? "↰":GOTO 700
750 ? "   INPUT N FOR NEXT 10 AFFILIAT
IONS":? "              INPUT Q TO QUIT"
760 ? "                WHICH";:INPUT K$
:IF K$="Q" THEN CLOSE #1:? "↰":GOTO 70
0
761 POSITION 0,3:FOR X=1 TO 10:? SP$;:
IF X<10 THEN ?
762 NEXT X
765 TRAP 780:POSITION 0,3:FOR X=1 TO 1
0:GOSUB 500:? CNO;". ";A$;" ";INT((VAL
(SC1$)+VAL(SC2$)+VAL(SC3$))/3);:IF X<1
0 THEN ?
766 NEXT X:POSITION 21,22:? " ":POSITI
ON 0,22:GOTO 760
780 POSITION 10,19:? "
    ":? "          ALL AFFILIATIONS DISPLAY
ED    ":? "          HIT RETURN TO CONTI
NUE":CLOSE #1
785 ? "■":OPEN #1,4,0,"K:":GET #1,X:CL
OSE #1:? "↰":GOTO 700
1000 ? "↰":POSITION 0,11:? "      INSE
RT THE PROGRAM DISKETTE ":? "
HIT RETURN WHEN READY":CLOSE #2
1010 OPEN #2,4,0,"K:":GET #2,X:TRAP 10
20:CLOSE #1:OPEN #1,4,0,"D:TRACK.BAS":
CLOSE #1:CLOSE #2:RUN "D:TRACK.BAS"
1020 TRAP 40000:CLOSE #1:? "■■↰":GOTO 1
000
2000 TRAP 2100:LPRINT :LPRINT :LPRINT
:LPRINT "      AFFILIATIONS      AVE. S
CORE":LPRINT "    ----------------------
-----------"
2010 CLOSE #1:OPEN #1,4,0,"D:AFFIL.DAT
"
2015 TRAP 2100:CLOSE #3:OPEN #3,8,0,"P
ARKS"
```

```
2020 ? "   INPUT N FOR NEXT 10 AFFILIA
TIONS":? "              INPUT Q TO QUIT"
2025 ? "                    WHICH";:INPUT K
$:IF K$="Q" THEN CLOSE #1:? "K":GOTO 7
00
2030 TRAP 2040:FOR X=1 TO 10:GOSUB 500
:? #3;CNO;". ";A$;" ";INT((VAL(SC1$)+V
AL(SC2$)+VAL(SC3$))/3)
2031 NEXT X:POSITION 21,22:? " ":POSIT
ION 0,22:GOTO 2025
2040 POSITION 10,19:? "
    ":? "       ALL AFFILIATIONS DISPLA
YED    ":? "          HIT RETURN TO CONT
INUE":CLOSE #1
2050 ? "D":OPEN #1,4,0,"K:":GET #1,X:C
LOSE #1:? "K":GOTO 700
2100 TRAP 40000:? "                RETURN P
RINTER ON":FOR ZZ=1 TO 250:NEXT ZZ:? "
K":GOTO 700
3000 TRAP 700:? "KENTER AFFILIATION #"
;:INPUT M
3001 CLOSE #1:OPEN #1,4,0,"D:AFFIL.DAT
":FOR X=1 TO M:GOSUB 500:NEXT X:CLOSE
#1:POKE 201,3
3009 TRAP 2100:LPRINT :LPRINT :LPRINT
:LPRINT "          AFFILIATION #";M;:L
PRINT :LPRINT
3010 LPRINT " 1. NAME"," ",A$:LPRINT "
 2. ACRONYM",,B$:LPRINT " 3. ADDRESS L
.1",C1$:LPRINT " 4. ADDRESS L.2",C2$
3015 LPRINT " 5. ADDRESS L.3",C3$
3020 LPRINT " 6. ZIP CODE",,D5$:LPRINT
 " 7. PHONE #",,PH$:LPRINT " 8. CONTAC
T NAME",CN$:LPRINT " 9. SCORE:",,,SC1$

3030 LPRINT "10. SCORE:",,,SC2$:LPRINT
 "11. SCORE:",,,SC3$
3040 ? "K":GOTO 700
```

# EMU TRACK MEET DATABASE – CREATE HEAT

```
10 DIM K$(1),NAME$(1000),N$(25),INFO(2
00,4),HEAT(12,8),LN(2,8),INVLN(2,8),H$
(8)
12 FOR X=1 TO 8:READ Y:LN(1,X)=Y:READ
Y:LN(2,X)=Y:NEXT X
14 FOR X=1 TO 8:INVLN(1,LN(1,X))=X:INV
LN(2,LN(2,X))=X:NEXT X
15 DIM SP$(39):SP$=" ":SP$(39)=" ":SP$
(2)=SP$
20 POSITION 0,0:FOR X=1 TO 23:? "■";:N
EXT X
50 POKE 752,1:? "■":SETCOLOR 2,0,0:POS
ITION 0,0:POKE 82,0:POKE 16,64:POKE 53
774,64:TRAP 40000
76 POSITION 0,8:? "       INSERT THE D
ATA DISKETTE":? "       HIT RETURN TO
 CONTINUE";:OPEN #1,4,0,"K:":GET #1,X
77 TRAP 80:CLOSE #1:OPEN #2,4,0,"D:TRA
CK.BAS":? "■■■":CLOSE #2:GOTO 76
80 ? "■";:TRAP 40000:? "
 HEAT #0":? "    ENTRANT              MA
RK   AFF. LANE ";
85 ? "──────────":? :? :? :? :? :? :? :? :? "─
──────────";
90 ? :? :? "      ENTER NUMBER TO SEED
 AN EVENT":? "         HIT RETURN FOR M
AIN MENU"
95 ? "                WHICH";:INPUT K
$:IF K$="" THEN GOTO 1000
100 OPEN #1,4,0,"D:PART.DAT":PO=1:PNO=
0:PO1=1:FOR X=1 TO 12:FOR XX=1 TO 6:HE
AT(X,XX)=0:NEXT XX:NEXT X
105 POSITION 0,14:? "******":? "
    NUMBER OF LANES";:INPUT LANES:IF L
ANES<>6 AND LANES<>8 THEN 105
107 V=(LANES=8)+2*(LANES=6):? "↑    .
....SEEDING IN PROGRESS.....    "
110 TRAP 125:FOR Z=1 TO 1000:GOSUB 500
:IF INIT THEN NAME$(PO,PO+LEN(N$)-1)=N
$
112 IF INIT THEN FOR ZZ=PO+LEN(N$) TO
PO+20:NAME$(ZZ,ZZ)=" ":NEXT ZZ:PO=PO+2
0
115 IF INIT THEN INFO(PO1,0)=PO1*20-19
:INFO(PO1,1)=PNO:INFO(PO1,2)=AF:INFO(P
O1,3)=GR:INFO(PO1,4)=MARK:PO1=PO1+1
120 NEXT Z
125 TRAP 40000:CLOSE #1:FOR X=PO1 TO 2
00:INFO(X,1)=0:NEXT X:NHEAT=INT((PO1-1
)/LANES+0.99)
130 FOR M=PO1-1 TO 2 STEP -1:SPOT=0:BI
G=0:FOR MM=1 TO M
135 IF INFO(MM,4)>BIG THEN BIG=INFO(MM
,4):SPOT=MM:FOR X=0 TO 4:INFO(200,X)=I
NFO(MM,X):NEXT X
140 NEXT MM:IF SPOT<>0 THEN FOR X=0 TO
 4:INFO(SPOT,X)=INFO(M,X):INFO(M,X)=IN
FO(200,X):NEXT X
145 NEXT M
150 FOR X=1 TO LANES:IF X/2=INT(X/2) T
HEN 175
155 FOR XX=1 TO NHEAT:IF INFO((X-1)*NH
EAT+XX,1)<>0 THEN HEAT(XX,X)=(X-1)*NHE
AT+XX
160 NEXT XX:NEXT X:GOTO 200
175 FOR XX=NHEAT TO 1 STEP -1:IF INFO(
(X-1)*NHEAT+(NHEAT-XX)+1,1)<>0 THEN HE
AT(XX,X)=(X-1)*NHEAT+(NHEAT-XX)+1
180 NEXT XX:NEXT X:GOTO 200
200 POSITION 0,14:? "******":? "
   HIT + TO VIEW NEXT HEAT":? "      H
IT - TO VIEW PREVIOUS HEAT"
205 ? "   HIT P TO PRINT ALL HEATS BY
LANE # ":? " HIT S TO SWAP PARTICIPANT
S 'A' AND 'B'"
210 ? "          HIT RETURN FOR MAIN MEN
U":? "                WHICH";:INPUT K
$:IF K$="" THEN GOTO 1000
215 IF K$="+" THEN IC=1:GOTO 400
220 IF K$="-" THEN IC=-1:GOTO 400
225 IF K$="S" THEN 300
230 IF K$<>"P" THEN 200
235 POSITION 0,14:? "******":? "
     SET UP PRINTER ":? "          HI
T RETURN WHEN READY"
237 OPEN #1,4,0,"K:":GET #1,X:CLOSE #1
:TRAP 280:OPEN #1,8,0,"P:"
240 FOR H=1 TO NHEAT:? #1:? #1:FOR X=1
 TO 35:? #1;" ";:NEXT X:? #1;"HEAT #";
H:? #1;"   PARTICIPENT","  LANE","AFF
IL",
242 ? #1;" GRADE","ENTERING MARK":FOR
X=1 TO 66:? #1;"-";:NEXT X:? #1
245 FOR X=1 TO LANES:IF HEAT(H,INVLN(V
,X))<>0 THEN GOTO 250
247 NEXT X:GOTO 260
250 ? #1;INFO(HEAT(H,INVLN(V,X)),1);"
";
252 ? #1;NAME$(INFO(HEAT(H,INVLN(V,X))
,0),INFO(HEAT(H,INVLN(V,X)),0)+19);" "
;:? #1;X,:? #1;INFO(HEAT(H,INVLN(V,X))
,2),
255 ? #1;INFO(HEAT(H,INVLN(V,X)),3),:?
 #1;INFO(HEAT(H,INVLN(V,X)),4):NEXT X:
GOTO 260
260 ? #1:? #1:NEXT H:CLOSE #1:GOTO 200

280 TRAP 40000:CLOSE #1:GOTO 200
300 POSITION 0,14:? "******":? "
  PARTICIPANT A'S NUMBER";:INPUT K1:?
"       PARTICIPANT B'S NUMBER";:INPUT
 K2
305 FOUND=0:FOR X=1 TO PO1:IF INFO(X,1
)=K1 THEN FOR Y=0 TO 4:INFO(200,Y)=INF
O(X,Y):NEXT Y:ANO=X:FOUND=1
307 NEXT X
310 FOR X=1 TO PO1:IF INFO(X,1)=K2 AND
 FOUND THEN FOR Y=0 TO 4:INFO(ANO,Y)=I
NFO(X,Y):INFO(X,Y)=INFO(200,Y):NEXT Y:
GOTO 320
315 NEXT X
320 POP :IC=0:GOTO 400
400 IF HNO+IC>NHEAT OR HNO+IC<1 THEN 2
00
405 HNO=HNO+IC:POSITION 21,0:? HNO;" "
:FOR X=1 TO LANES:IF HEAT(HNO,X)<>0 TH
EN GOTO 409
407 POSITION 0,X+2:? "
                      ";:NEXT X:GOTO
200
409 POSITION 0,X+2:? INFO(HEAT(HNO,X),
1);" "
410 POSITION 2,X+2:? NAME$(INFO(HEAT(H
NO,X),0),INFO(HEAT(HNO,X),0)+19):POSIT
ION 23,X+2:? INFO(HEAT(HNO,X),4);"  "
415 POSITION 31,X+2:? INFO(HEAT(HNO,X)
,2);" ":POSITION 36,X+2:? LN(V,X):NEXT
 X:GOTO 200
500 INIT=0:INPUT #1;N$:INPUT #1;H$:AF=
VAL(H$):INPUT #1;H$:GR=VAL(H$):INPUT #
1;H$
505 FOR X=1 TO 6:INPUT #1;H$:IF VAL(H$
)=VAL(K$) THEN INIT=1:HERE=X
510 NEXT X
515 FOR X=1 TO 6:INPUT #1;H$:IF INIT A
ND X=HERE THEN MARK=VAL(H$)
520 NEXT X:PNO=PNO+1:RETURN
1000 ? "■":POSITION 9,10:? "INSERT PRO
GRAM DISKETTE":? "          HIT RETURN
 WHEN READY":CLOSE #1
1005 OPEN #1,4,0,"K:":GET #1,X:CLOSE #
1
1010 TRAP 1020:OPEN #1,4,0,"D:TRACK.BA
S":CLOSE #1:RUN "D:TRACK.BAS"
1020 TRAP 40000:CLOSE #1:? "■■■":GOTO 1
000
5000 DATA 4,4,5,5,6,3,3,6,2,2,7,1,1,0,
8,0
```

# EMU TRACK MEET DATABASE – EVENTS

```
10 DIM A$(25),B$(3),C$(3),D$(3),E$(25)
,F$(10),5$(25),H$(10),I$(25),K$(2),T$(
1),G$(25),G1$(3),G2$(10):FLAG=0
15 DIM J$(10),N$(3),SP$(39),G3$(5),G4$
(6):SP$=" ":SP$(39)=" ":SP$(2)=SP$
20 POSITION 0,0:FOR X=1 TO 23:? "■";:N
EXT X
50 POKE 752,1:? "ß":SETCOLOR 2,0,0:POS
ITION 0,0:POKE 82,0:POKE 16,64:POKE 53
774,64:TRAP 40000
76 POSITION 0,8:? "       INSERT THE D
ATA DISKETTE":? "       HIT RETURN TO
 CONTINUE";:OPEN #1,4,0,"K:":GET #1,X
77 TRAP 80:CLOSE #1:OPEN #2,4,0,"D:TRA
CK.BAS":? "■■ß":CLOSE #2:GOTO 76
80 ? "ß":TRAP 40000:POSITION 0,11:? "
  HIT E TO EDIT/VIEW EXISTING EVENT":?
 "       HIT N TO ADD NEW EVENT"
81 ? "       HIT L TO LIST/PRINT EVENTS
"
84 TRAP 80:? "                  WHICH ←
←";:INPUT T$:IF T$="E" THEN ? :? "  CO
DE NUMBER OF DESIRED EVENT";:INPUT M
85 TRAP 40000:IF T$<>"E" AND T$<>"N" A
ND T$<>"L" THEN ? "↑↑":GOTO 84
86 TRAP 80:IF T$="L" THEN CLOSE #1:OPE
N #1,4,0,"D:EVENT.DAT":? "ß":GOTO 700
87 TRAP 80:IF T$="E" THEN CLOSE #1:OPE
N #1,4,0,"D:EVENT.DAT":CNO=0:FOR X=1 T
O M:GOSUB 500:NEXT X:CLOSE #1:GOTO 99
89 TRAP 91:CLOSE #1:OPEN #1,4,0,"D:EVE
NT.DAT"
90 TRAP 92:CNO=0:FOR X=1 TO 1000:GOSUB
 500:NEXT X
91 TRAP 40000:CLOSE #1:OPEN #1,8,0,"D:
EVENT.DAT":FLAG=1:CLOSE #1:GOTO 89
92 TRAP 40000:CLOSE #1:CNO=CNO+1:A$=""
:B$="":C$="":D$="":E$="":F$="":5$="":H
$="":I$="":J$="":N$=""
99 ? "ß":TRAP 40000:? "          EVENT
 #";CNO;:? :? :POKE 201,3
100 ? " 1. EVENT",A$:? " 2. MEASUREMEN
T TYPE",,B$:? " 3. SEEDING CODE",,,,C$
:? " 4. DIVISION",,,,,D$
102 ? " 5. PRELIM:";" ";E$:? " 6. TIME
 OF PRELIM5",,,F$:? " 7. SEMI:";"   ";
5$
105 ? " 8. TIME OF SEMI5",,,H$:? " 9.
FINAL:";"  ";I$:? "10. TIME OF FINAL5"
,,,J$:? "11. NUMBER TO FORCE HEAT5",N$

125 TRAP 99:? :? "       HIT A TO ENTE
R ALL ITEM5":? "   ENTER NUMBER TO EDI
T SINGLE ITEM"
130 ? "       HIT S TO SAVE DATA ON DIS
K":? "       HIT RETURN FOR MAIN MENU"
:? "                WHICH";:INPUT K$:?
 "ß"
135 IF K$="" THEN CLOSE #1:GOTO 1000
137 IF K$="A" THEN ALL=1:GOTO 140
138 ALL=0
140 IF K$="1" OR ALL THEN ? :? "ENTER
EVENT NAME";:INPUT A$
150 IF K$="2" OR ALL THEN ? :? "ENTER
MEASUREMENT TYPE":INPUT B$
160 IF K$="3" OR ALL THEN ? :? "ENTER
SEEDING CODE":INPUT C$
170 IF K$="4" OR ALL THEN ? :? "ENTER
DIVISION":INPUT D$
180 IF K$="5" OR ALL THEN ? :? "ENTER
DAY OF PRELIM5":INPUT E$
190 IF K$="6" OR ALL THEN ? :? "ENTER
TIME OF PRELIM5":INPUT F$
200 IF K$="7" OR ALL THEN ? :? "ENTER
DAY OF SEMI5":INPUT 5$
202 IF K$="8" OR ALL THEN ? :? "ENTER
TIME OF SEMI5":INPUT H$
205 IF K$="9" OR ALL THEN ? :? "ENTER
DAY OF FINAL5":INPUT I$
210 IF K$="10" OR ALL THEN ? :? "ENTER
 TIME OF FINAL5":INPUT J$
215 IF K$="11" OR ALL THEN ? :? "ENTER
 NUMBER TO FORCE HEAT5":INPUT N$
225 IF K$="5" THEN 300
250 ? "ß":GOTO 99
300 ? " ARE YOU SURE THE ENTRIES ARE C
ORRECT";:INPUT K$:IF K$<>"Y" AND K$<>"
YES" THEN 99
301 IF T$="N" AND FLAG THEN CLOSE #1:O
PEN #1,8,0,"D:EVENT.DAT":FLAG=0:GOTO 3
06
302 IF T$="N" AND  NOT FLAG THEN CLOSE
 #1:OPEN #1,9,0,"D:EVENT.DAT":GOTO 306

303 IF T$="E" AND M=1 THEN CLOSE #1:OP
EN #1,12,0,"D:EVENT.DAT":GOTO 306
305 CLOSE #1:OPEN #1,12,0,"D:EVENT.DAT
":FOR X=1 TO M-1:GOSUB 520:NEXT X
306 IF LEN(A$)=25 THEN 308
307 FOR Y=LEN(A$)+1 TO 25:A$(Y,Y)=" ":
NEXT Y
308 IF LEN(B$)=3 THEN 310
309 FOR Y=LEN(B$)+1 TO 3:B$(Y,Y)=" ":N
EXT Y
310 IF LEN(C$)=3 THEN 312
311 FOR Y=LEN(C$)+1 TO 3:C$(Y,Y)=" ":N
EXT Y
312 IF LEN(D$)=3 THEN 314
313 FOR Y=LEN(D$)+1 TO 3:D$(Y,Y)=" ":N
EXT Y
314 IF LEN(E$)=25 THEN 316
315 FOR Y=LEN(E$)+1 TO 25:E$(Y,Y)=" ":
NEXT Y
316 IF LEN(F$)=10 THEN 318
317 FOR Y=LEN(F$)+1 TO 10:F$(Y,Y)=" ":
NEXT Y
318 IF LEN(5$)=25 THEN 320
319 FOR Y=LEN(5$)+1 TO 25:5$(Y,Y)=" ":
NEXT Y
320 IF LEN(H$)=10 THEN 322
321 FOR Y=LEN(H$)+1 TO 10:H$(Y,Y)=" ":
NEXT Y
322 IF LEN(I$)=25 THEN 324
323 FOR Y=LEN(I$)+1 TO 25:I$(Y,Y)=" ":
NEXT Y
324 IF LEN(J$)=10 THEN 326
325 FOR Y=LEN(J$)+1 TO 10:J$(Y,Y)=" ":
NEXT Y
326 IF LEN(N$)=3 THEN 330
327 FOR Y=LEN(N$)+1 TO 3:N$(Y,Y)=" ":N
EXT Y
330 PRINT #1;A$:PRINT #1;B$:PRINT #1;C
$:PRINT #1;D$:PRINT #1;E$:PRINT #1;F$:
PRINT #1;5$:PRINT #1;H$
331 PRINT #1;I$:PRINT #1;J$:PRINT #1;N
$:CLOSE #1:GOTO 80
500 INPUT #1;A$:INPUT #1;B$:INPUT #1;C
$:INPUT #1;D$:INPUT #1;E$:INPUT #1;F$:
INPUT #1;5$:INPUT #1;H$
515 INPUT #1;I$:INPUT #1;J$:INPUT #1;N
$:CNO=CNO+1:RETURN
520 INPUT #1;G$:INPUT #1;G1$:INPUT #1;
G1$:INPUT #1;G1$:INPUT #1;G$:INPUT #1;
G2$:INPUT #1;G$:INPUT #1;G2$
525 INPUT #1;G$:INPUT #1;G2$:INPUT #1;
G1$:RETURN
700 CLOSE #1:OPEN #1,4,0,"D:EVENT.DAT"

701 ? "ß":CNO=0:? "       EVENTS      TI
ME OF FINAL5    ":? "      ___________
_____"
705 POSITION 0,13:? "      ___________
_____":? :? "          HIT
D TO DISPLAY EVENTS"
710 ? "       HIT P TO PRINT LIST OF EVE
NTS":? "    HIT A TO PRINT ALL INFO. ON
 EVENTS";
715 ? "              HIT RETURN FOR MAIN
MENU":? "                  WHICH";:INPUT
K$
720 IF K$="" THEN CLOSE #1:GOTO 1000
725 IF K$="P" THEN 2000
726 IF K$="A" THEN 3000
730 IF K$<>"D" THEN ? "ß":GOTO 700
750 ? "          INPUT N FOR NEXT 10 EVEN
T5":? "              INPUT Q TO QUIT"
760 ? "                    WHICH";:INPUT K$
:IF K$="Q" THEN CLOSE #1:? "ß":GOTO 70
0
761 POSITION 0,3:FOR X=1 TO 10:? SP$;:
IF X<10 THEN ?
762 NEXT X
765 TRAP 780:POSITION 0,3:FOR X=1 TO 1
0:GOSUB 500:? CNO;".";A$;" ";J$;:IF X<
10 THEN ?
766 NEXT X:POSITION 21,22:? " ":POSITI
ON 0,22:GOTO 760
780 POSITION 10,19:? "
   ":? "              ALL EVENTS DISPLAYED
    ":? "              HIT RETURN TO CONTIN
UE":CLOSE #1
785 ? "■":OPEN #1,4,0,"K:":GET #1,X:CL
OSE #1:? "ß":GOTO 700
1000 ? "ß":POSITION 0,11:? "      INSE
RT THE PROGRAM DISKETTE ":? "
HIT RETURN WHEN READY":CLOSE #2
1010 OPEN #2,4,0,"K:":GET #2,X:TRAP 10
20:CLOSE #1:OPEN #1,4,0,"D:TRACK.BA5":
CLOSE #1:CLOSE #2:RUN "D:TRACK.BA5"
1020 TRAP 40000:CLOSE #1:? "■■ß":GOTO 1
000
2000 TRAP 2100:LPRINT :LPRINT :LPRINT
:LPRINT "     EVENTS          TIME OF
FINAL5":LPRINT "    ----------------
--------------"
2010 CLOSE #1:OPEN #1,4,0,"D:EVENT.DAT
"
2015 TRAP 2100:CLOSE #3:OPEN #3,8,0,"P
ARK5"
2020 ? "       INPUT N FOR NEXT 10 EVE
```

```
NTS":? "            INPUT Q TO QUIT"
2025 ? "                  WHICH";:INPUT K
$:IF K$="Q" THEN CLOSE #1:? "R":GOTO 7
00
2030 TRAP 2040:FOR X=1 TO 10:GOSUB 500
:? #3;CNO;". ";A$;" ";J$
2031 NEXT X:POSITION 21,22:? " ":POSIT
ION 0,22:GOTO 2025
2040 POSITION 10,19:? "
    ":? "           ALL EVENTS DISPLAYE
D    ":? "          HIT RETURN TO CONTI
NUE":CLOSE #1
2050 ? "Q":OPEN #1,4,0,"K:":GET #1,X:C
LOSE #1:? "R":GOTO 700
2100 TRAP 40000:? "              RETURN P
RINTER ON":FOR ZZ=1 TO 250:NEXT ZZ:? "
R":GOTO 700
3000 TRAP 700:? "RENTER EVENT #";:INPU
T M
3001 CLOSE #1:OPEN #1,4,0,"D:EVENT.DAT
":FOR X=1 TO M:GOSUB 500:NEXT X:CLOSE
#1:POKE 201,3
3009 TRAP 2100:LPRINT :LPRINT :LPRINT
:LPRINT "           EVENT #";M;:LPRINT
:LPRINT
3010 LPRINT " 1. EVENT",A$:LPRINT " 2.
 MEASUREMENT TYPE",,B$:LPRINT " 3. SEE
DING CODE",,,,C$:LPRINT " 4. DIVISION"
,,,,,D$
3020 LPRINT " 5. DAY OF PRELIMS",,,E$:
LPRINT " 6. TIME OF PRELIMS",,,F$:LPRI
NT " 7. DAY OF SEMIS",,,,S$
3025 LPRINT " 8. TIME OF SEMIS",,,H$:L
PRINT " 9. DAY OF FINALS",,,I$:LPRINT
"10. TIME OF FINALS",,,J$
3027 LPRINT "11. NUMBER TO FORCE HEATS
",N$
3040 ? "R":GOTO 700
```

# EMU TRACK MEET DATABASE – PARTICIPANTS

```
5 DIM G(6),K$(1),A$(17),T$(1),C$(10),O
$(8),R$(1),N$(17),SP$(39),N(6),EM(6),H
$(8),H1$(8)
10 SP$=" ":SP$(39)=" ":SP$(2)=SP$
20 POSITION 0,0:FOR X=1 TO 23:? "▮";":
N.X":P=1
50 POKE 752,1:? "▮":SETCOLOR 2,0,0:POS
ITION 0,0:POKE 82,0:POKE 16,64:POKE 53
774,64:TRAP 40000
76 POSITION 0,8:? "       INSERT THE D
ATA DISKETTE":? "      HIT RETURN TO
 CONTINUE";:OPEN #1,4,0,"K:":GET #1,X
77 TRAP 80:CLOSE #1:OPEN #2,4,0,"D:TRA
CK.BAS":? "▮▮▮":CLOSE #2:GOTO 76
80 TRAP 40000:? "▮":POSITION 0,11:? "H
IT E TO EDIT/VIEW EXISTING PARTICIPANT
S↑":? "      HIT N TO ADD NEW PARTICIPA
NT"
81 ? "   HIT L TO LIST/PRINT PARTICIPA
NTS"
84 TRAP 80:? "                WHICH ←
←";:INPUT T$:IF T$="E" THEN ? :? "ENTE
R PARTICIPANT NUMBER";:INPUT M
85 IF T$<>"E" AND T$<>"N" AND T$<>"L"
THEN ? "↑↑":GOTO 84
86 TRAP 80:IF T$="L" THEN CLOSE #1:OPE
N #1,4,0,"D:PART.DAT":? "▮":GOTO 500
96 IF M=1 AND T$="E" THEN CLOSE #2:OPE
N #2,12,0,"D:PART.DAT":GOSUB 7000:GOTO
 99
97 IF T$="E" THEN CLOSE #2:OPEN #2,12,
0,"D:PART.DAT":FOR L=1 TO M:GOSUB 7000
:NEXT L
98 TRAP 10000:IF T$="N" THEN CLOSE #2:
OPEN #2,9,0,"D:PART.DAT":GOSUB 10500
99 TRAP 40000:? "▮":IF T$="E" THEN ? "
ENTRANT #";M
100 ? "1. ENTRANT NAME",A$:? "2. AFFIL
IATION #",B:? "3. SCHOOL GRADE",G:? "4
. EVENT #",C
105 ? "5. EVENTS ENTERED",:FOR O=1 TO
6:? G(O);" ";:NEXT O:?
110 ? :? "  ENTERING MARKS":FOR O=1 T
O 6:IF G(O)<>0 THEN ? "   EVENT #";G(O
),EM(O):NEXT O
115 POP
180 TRAP 95:? :? :? "       HIT A TO E
NTER ALL ITEMS":? "   ENTER NUMBER TO
EDIT SINGLE ITEM"
190 ? "       HIT S TO SAVE DATA ON DIS
K":? "       HIT RETURN FOR MAIN MENU"
:? "              WHICH";:INPUT K$:?
 "▮"
200 IF K$="" THEN CLOSE #2:GOTO 1000
203 IF K$="S" THEN 700
205 IF K$="A" THEN ALL=1:TRAP 99:GOTO
210
206 ALL=0:TRAP 99
210 IF K$="1" OR ALL THEN ? :? "ENTER
ENTRANT NAME":INPUT A$
220 IF K$="2" OR ALL THEN ? :? "ENTER
AFFILIATION #":INPUT B
230 IF K$="3" OR ALL THEN ? :? "ENTER
SCHOOL GRADE":INPUT G
235 IF K$="4" OR ALL THEN ? :? "ENTER
EVENT #":INPUT C
250 IF K$="5" OR ALL THEN ? :? "ENTER
EVENTS ENTERED":? "ENTER ONE LINE AT A
 TIME AND BY EVENT #":REM ASK WHAT EVE
NTS
255 IF K$="5" OR ALL THEN GOSUB 5000
260 ? "▮":GOTO 100
500 CLOSE #2:OPEN #2,4,0,"D:PART.DAT"
501 ? "▮":CNO=0:? "      PARTICIPANT
   EVENTS ENTERED":? "
"
502 POSITION 0,13:? "
":? :? "      HIT D T
O DISPLAY PARTICIPANTS"
504 ? "  HIT P TO PRINT LIST OF PARTIC
IPANTS":? "HIT A TO PRINT ALL INFO. ON
 PARTICIPANTS";
505 ? "       HIT RETURN FOR MAIN MENU
":? "              WHICH";:INPUT K$
506 IF K$="" THEN CLOSE #2:GOTO 1000
507 IF K$="P" THEN 2000
508 IF K$="A" THEN 3000
509 IF K$<>"D" THEN ? "▮":GOTO 500
510 CLOSE #2:OPEN #2,4,0,"D:PART.DAT"
512 ? "  INPUT N FOR NEXT 10 PARTICIP
ANTS":? "            INPUT Q TO QUIT"
514 ? "              WHICH";:INPUT K$
:IF K$="Q" THEN CLOSE #2:? "▮":GOTO 50
0
515 POSITION 0,3:FOR X=1 TO 10:? SP$;:
IF X<10 THEN ?
517 NEXT X
520 TRAP 522:POSITION 0,3:FOR X=1 TO 1
0:GOSUB 7000:? CNO;". ";A$;" ";:FOR Y=
1 TO 6:? G(Y);" ";:NEXT Y:IF X<10 THEN
 ?
521 NEXT X:POSITION 21,22:? " ":POSITI
ON 0,22:GOTO 514
522 POSITION 10,19:? "
    ":? "      ALL PARTICIPANTS DISPL
AYED    ":? "       HIT RETURN TO CON
TINUE"
525 ? "▮":CLOSE #1:OPEN #1,4,0,"K:":GE
T #1,X:CLOSE #1:? "▮":GOTO 500
700 ? "▮":? "ARE YOU SURE OF CORRECT E
NTRIES";:INPUT R$
701 IF R$<>"Y" THEN 99
702 IF T$="E" AND M<>1 THEN CLOSE #2:O
PEN #2,12,0,"D:PART.DAT":FOR ML=1 TO M
-1:GOSUB 8000:NEXT ML:GOTO 710
703 IF T$="E" THEN CLOSE #2:OPEN #2,12
,0,"D:PART.DAT"
706 IF LEN(A$)=17 THEN 710
707 FOR L=LEN(A$)+1 TO 17:A$(L,L)=" ":
NEXT L
710 ? #2;A$
711 H1$=STR$(B):GOSUB 9000:H$(9-LEN(H1
$),8)=H1$:? #2;H$
712 H1$=STR$(G):GOSUB 9000:H$(9-LEN(H1
$),8)=H1$:? #2;H$
713 H1$=STR$(C):GOSUB 9000:H$(9-LEN(H1
$),8)=H1$:? #2;H$
715 FOR O=1 TO 6:H1$=STR$(G(O)):GOSUB
9000:H$(9-LEN(H1$),8)=H1$:? #2;H$:NEXT
 O
717 FOR O=1 TO 6:H1$=STR$(EM(O)):GOSUB
 9000:H$(9-LEN(H1$),8)=H1$:? #2;H$:NEX
T O
720 CLOSE #2:GOTO 80
1000 ? "▮":POSITION 0,11:? "      INSE
RT THE PROGRAM DISKETTE ":? "
HIT RETURN WHEN READY":CLOSE #2
1010 OPEN #2,4,0,"K:":GET #2,X:TRAP 10
20:OPEN #1,4,0,"D:TRACK.BAS":CLOSE #1:
CLOSE #2:RUN "D:TRACK.BAS"
1020 TRAP 40000:CLOSE #1:? "▮▮":GOTO 1
000
2000 TRAP 2100:LPRINT "     PARTICIPAN
T      EVENTS ENTERED":LPRINT "    ----
-------------------------"
2010 CLOSE #2:OPEN #2,4,0,"D:PART.DAT"
2015 TRAP 2100:CLOSE #3:OPEN #3,8,0,"P
ARKS"
2020 ? "  INPUT N FOR NEXT 10 PARTICI
PANTS":? "            INPUT Q TO QUIT"
2025 ? "              WHICH";:INPUT K
$:IF K$="Q" THEN CLOSE #2:? "▮":GOTO 5
00
2030 TRAP 2040:FOR X=1 TO 10:GOSUB 700
0:? #3;CNO;". ";A$;" ";:FOR Y=1 TO 6:?
 #3;G(Y);" ";:NEXT Y:? #3
2036 NEXT X:POSITION 21,22:? " ":POSIT
ION 0,22:GOTO 2025
2040 POSITION 10,19:? "
    ":? "      ALL PARTICIPANTS DISPLA
YED    ":? "       HIT RETURN TO CONT
INUE":CLOSE #2
2050 ? "▮":CLOSE #3:CLOSE #1:OPEN #1,4
,0,"K:":GET #1,X:CLOSE #1:? "▮":GOTO 5
00
2100 TRAP 40000:? :? "          ▮▮TUR
N PRINTER ON":FOR ZZ=1 TO 250:NEXT ZZ:
? "▮":GOTO 500
3000 TRAP 500:? "▮ENTER PARTICIPANT #"
;:INPUT M
3001 CLOSE #2:OPEN #2,4,0,"D:PART.DAT"
:FOR X=1 TO M:GOSUB 7000:NEXT X:CLOSE
#2
3009 TRAP 2100:LPRINT "          PARTI
CIPANT #";M;:LPRINT :LPRINT
3010 LPRINT "1. ENTRANT NAME",A$:LPRIN
T "2. AFFILIATION #",B:LPRINT "3. SCHO
OL GRADE",G:LPRINT "4. EVENT #",C
3020 OPEN #3,8,0,"P:":? #3;"5. EVENTS
ENTERED",:FOR O=1 TO 6:? #3;G(O);" ";:
NEXT O:? #3
3025 ? #3:? #3;"ENTERING MARKS":FOR O=
1 TO 6:IF G(O)<>0 THEN ? #3;"EVENT #";
G(O),EM(O):NEXT O
3026 POP :CLOSE #3
3040 ? "▮":GOTO 500
5000 P=1:? "ENTER EVENT # OR RETURN TO
 CONTINUE"
5002 ? "EVENT #";:INPUT O$:IF O$="" TH
EN FOR V=P TO 6:G(V)=0:EM(V)=0:NEXT V:
RETURN
5003 G(P)=VAL(O$)
5004 ? "ENTERING MARK FOR EVENT #";O$;
:INPUT O$:IF O$="" THEN GOTO 5004
5006 EM(P)=VAL(O$):P=P+1:IF P>6 THEN R
ETURN
5007 GOTO 5002
7000 INPUT #2;A$:INPUT #2;H$:B=VAL(H$)
:INPUT #2;H$:G=VAL(H$):INPUT #2;H$:C=V
```

```
AL(H$)
7003 FOR I=1 TO 6:INPUT #2;H$:G(I)=VAL
(H$):NEXT I
7005 FOR I=1 TO 6:INPUT #2;H$:EM(I)=VA
L(H$):NEXT I:CNO=CNO+1
7010 RETURN
8000 FOR I=1 TO 16:INPUT #2;N$:NEXT I
9000 H$="00000000":RETURN
10000 TRAP 98:CLOSE #4:OPEN #4,8,0,"D:
PART.DAT":CLOSE #4:GOTO 98
10500 A$=" ":B=0:G=0:C=0:C$=" ":FOR I=
1 TO 6:G(I)=0:EM(I)=0:NEXT I:RETURN
```

# EMU TRACK MEET DATABASE – RESULTS

```
10 DIM T$(2),H$(20),NAME$(1000),K$(2),
AF(6),PNTS(6),PART(2,6),R$(1),K1$(2),K
2$(2),EN$(2),SCORE(30),H1$(8),H2$(20),
CC(2,50)
11 DIM CF(50)
25 POSITION 0,0:FOR X=1 TO 22:? "■";:N
EXT X
50 POKE 752,1:SETCOLOR 2,0,0:POSITION
0,0:POKE 82,0:POKE 16,64:POKE 53774,64
:TRAP 40000
76 POSITION 0,8:? "       INSERT THE D
ATA DISKETTE":? "          HIT RETURN TO
 CONTINUE";:OPEN #1,4,0,"K:":GET #1,X
77 TRAP 78:CLOSE #1:OPEN #2,4,0,"D:TRA
CK.BAS":? "■■■":CLOSE #2:GOTO 76
78 TRAP 40000:OPEN #1,4,0,"D:MEET.DAT"
:FOR X=1 TO 4:INPUT #1;H$:NEXT X:FOR X
=1 TO 3:INPUT #1;Y:NEXT X:FOR X=1 TO 6

79 INPUT #1;Y:PNTS(X)=Y:NEXT X:CLOSE #
1
80 ? "■";:? "                    EVENT #0"

85 ? "PLACE      PARTICIPANT      AFFIL
  POINTS";:? "■■■
```

```
449 NEXT Y
452 OPEN #1,4,0,"D:PART.DAT":X1=STA:FO
R X=1 TO CC(2,LA):IF X<>CC(2,X1) THEN
FOR Z=1 TO 16:INPUT #1;H$:NEXT Z:NEXT
X
453 INPUT #1;H$:GOSUB 159:NAME$(CC(1,X
1)*20-19,CC(1,X1)*20)=H$:INPUT #1;H$:C
F(CC(1,X1))=VAL(H$):X1=X1+1
454 FOR Z=1 TO 14:INPUT #1;H$:NEXT Z:N
EXT X:CLOSE #1:IF STA=1 THEN OPEN #1,8
,0,"D:CROSS.DAT"
455 IF STA>1 THEN OPEN #1,9,0,"D:CROSS
.DAT"
456 FOR Y=STA TO LA
458 H$=NAME$(Y*20-19,Y*20):K1$="00":K2
$=STR$(CF(Y)):K1$(3-LEN(K2$),2)=K2$:PR
INT #1;H$:PRINT #1;K1$:NEXT Y
459 GOTO 400
475 HN=1:TRAP 490:OPEN #1,4,0,"D:CROSS
.DAT"
476 POSITION 0,12:? "[10 inverse]":? "
       HIT [N] TO VIEW NEXT 6 ":? "
       HIT [Q] TO QUIT":? "
   WHICH";
477 INPUT K$:IF K$="Q" THEN 490
478 IF K$<>"N" THEN 476
479 FOR X=3 TO 8:POSITION 1,X:? "
                                ":NEXT X

480 POSITION 0,3:FOR X=HN TO HN+5:? "
 ";X;"   ";:INPUT #1;H$:GOSUB 159:? H$
;"   ";:INPUT #1;H$:? VAL(H$):NEXT X:H
N=HN+6:GOTO 476
490 POSITION 0,12:? "[10 inverse]":TRAP
40000:CLOSE #1:? "          HIT [RETURN] F
OR C-C MENU":OPEN #1,4,0,"K:":GET #1,X

492 FOR X=3 TO 8:POSITION 1,X:? "
                                ":NEXT X
:CLOSE #1:GOTO 400
500 INPUT #1;EN$:EN=VAL(EN$):FOR Z=1 T
O 6:INPUT #1;H$:NAME$(Z*20-19,Z*20)=H$

505 INPUT #1;H$:AF(Z)=VAL(H$):NEXT Z:I
N=IN+1:RETURN
510 FOR Q=1 TO 13:INPUT #1;H$:NEXT Q:R
ETURN
600 POSITION 23,0:? EN
602 POSITION 0,3:FOR Z=1 TO 6:? "  ";Z
;"   ";NAME$(Z*20-19,Z*20);"   ";AF(Z)
;"      ";PNT$(Z);" ":NEXT Z:RETURN
900 TRAP 930:POSITION 0,12:? "[8 inverse
]":? "                TURN ON PRINTER":?
"       HIT [RETURN] WHEN READY"
910 OPEN #1,4,0,"K:":GET #1,X:CLOSE #1
:CLOSE #2:OPEN #1,4,0,"D:CROSS.DAT":OP
EN #2,8,0,"P:"
915 ? #2;"PLACE   PARTICIPANT       AFF
IL":? #2;"-----------------------------
-----"
917 FOR X=1 TO 50:INPUT #1;H$:GOSUB 15
9:INPUT #1;H1$:? #2;"  ";X;"   ";H$;"
";VAL(H1$):NEXT X
930 CLOSE #1:CLOSE #2:GOTO 400
1000 ? "[CLR]":POSITION 0,11:? "      INSE
RT THE PROGRAM DISKETTE ":? "
HIT [RETURN] WHEN READY":CLOSE #2
1010 OPEN #2,4,0,"K:":GET #2,X:TRAP 10
20:OPEN #1,4,0,"D:TRACK.BAS":CLOSE #1:
CLOSE #2:RUN "D:TRACK.BAS"
1020 TRAP 40000:CLOSE #1:? "[CLR]":GOTO 1
000
```

# EMU TRACK MEET DATABASE – START NEW MEET

```
10 DIM A$(19),B$(19),C$(10),D$(19),H(6
),K$(1),T$(1),NAME$(14)
25 POSITION 0,0:FOR X=1 TO 22:? "■";:N
EXT X
50 POKE 752,1:SETCOLOR 2,0,0:POSITION
0,0:POKE 82,0:POKE 16,64:POKE 53774,64
:TRAP 40000
75 FOR I=1 TO 6:H(I)=0:NEXT I
76 POSITION 0,8:? "       INSERT THE D
ATA DISKETTE":? "          HIT RETURN TO
 CONTINUE";:OPEN #1,4,0,"K:":GET #1,X
77 TRAP 80:CLOSE #1:OPEN #2,4,0,"D:TRA
CK.BAS":? "■■":CLOSE #2:GOTO 76
80 TRAP 40000:? "■":POSITION 0,11:? "
  HIT E TO EDIT/VIEW EXISTING MEET":?
"     HIT N TO START NEW MEET DISK"
81 ? "                    WHICH  ←←";:INPU
T T$
82 IF T$<>"E" AND T$<>"N" THEN ? "↑↑":
GOTO 81
84 TRAP 90:IF T$="E" THEN GOSUB 500:GO
TO 99
85 TRAP 95:OPEN #1,4,0,"D:MEET.DAT":CL
OSE #1:? :? "     USE E TO EDIT EXISTI
NG MEET":FOR X=1 TO 350:NEXT X:GOTO 80

90 TRAP 40000:? :? "    USE N TO START
 A NEW MEET DISK":FOR X=1 TO 350:NEXT
X:GOTO 80
95 TRAP 40000
99 ? "■":TRAP 40000:? "         GENERAL
 MEET INFORMATION":? :?
100 ? " 1. MEET NAME",A$:? " 2. MEET L
OCATION",B$:? " 3. FIRST DAY",C$
105 ? " 4. FIRST DATE",D$:? " 5. NUMBE
R OF DAYS",E:? " 6. NUMBER OF LANES ";
F
110 ? " 7. SCORING FORMULA ";:FOR I=1
TO 6:? H(I);" ";:NEXT I:? :? " 8. FIRS
T ID NUMBER ";ID
125 TRAP 100:? :? :? "       HIT A TO
ENTER ALL ITEMS":? "   ENTER NUMBER TO
 EDIT SINGLE ITEM"
130 ? "       HIT S TO SAVE DATA ON DIS
K":? "       HIT RETURN FOR MAIN MENU"
:? "                WHICH";:INPUT K$:?
 "■"
135 IF K$="" THEN CLOSE #1:GOTO 1000
137 IF K$="A" THEN ALL=1:GOTO 140
138 ALL=0
140 IF K$="1" OR ALL THEN ? :? "ENTER
MEET NAME":INPUT A$
150 IF K$="2" OR ALL THEN ? :? "ENTER
MEET LOCATION":INPUT B$
160 IF K$="3" OR ALL THEN ? :? "ENTER
FIRST DAY":INPUT C$
170 IF K$="4" OR ALL THEN ? :? "ENTER
FIRST DATE":INPUT D$
180 IF K$="5" OR ALL THEN ? :? "ENTER
NUMBER OF DAYS":INPUT E
190 IF K$="6" OR ALL THEN ? :? "ENTER
NUMBER OF LANES":INPUT F
205 IF K$="7" OR ALL THEN ? :? "ENTER
SCORING FORMULA":? "I.E. 1ST 10, 2ND 8
, 3RD 6, 4TH 4, 5TH 2, 6TH 1"
210 IF K$="7" OR ALL THEN ? :FOR I=1 T
O 6:? "ENTER SCORE FOR ";I;"TH PLACE":
INPUT H:H(I)=H:NEXT I
215 IF K$="8" OR ALL THEN ? :? "ENTER
FIRST ID NUMBER":INPUT ID
225 IF K$="S" THEN 300
250 ? "■":GOTO 99
300 TRAP 306:? " ARE YOU SURE THE ENTR
IES ARE CORRECT";:INPUT K$:IF K$<>"Y"
AND K$<>"YES" THEN 99
305 CLOSE #1:OPEN #1,8,0,"D:MEET.DAT"
306 IF LEN(A$)=19 THEN 308
307 FOR Y=LEN(A$)+1 TO 19:A$(Y,Y)=" ":
NEXT Y
308 IF LEN(B$)=19 THEN 310
309 FOR Y=LEN(B$)+1 TO 19:B$(Y,Y)=" ":
NEXT Y
310 IF LEN(C$)=10 THEN 312
311 FOR Y=LEN(C$)+1 TO 10:C$(Y,Y)=" ":
NEXT Y
312 IF LEN(D$)=19 THEN 320
313 FOR Y=LEN(D$)+1 TO 19:D$(Y,Y)=" ":
NEXT Y
320 PRINT #1;A$:PRINT #1;B$:PRINT #1;C
$:PRINT #1;D$
321 PRINT #1;E:PRINT #1;F:PRINT #1;ID
322 FOR X=1 TO 6:PRINT #1;H(X):NEXT X:
CLOSE #1:GOTO 80
500 OPEN #1,4,0,"D:MEET.DAT":INPUT #1;
A$:INPUT #1;B$:INPUT #1;C$:INPUT #1;D$

515 INPUT #1;E:INPUT #1;F:INPUT #1;ID
520 FOR X=1 TO 6:INPUT #1;H:H(X)=H:NEX
T X:CLOSE #1:RETURN
1000 ? "■":POSITION 0,11:? "      INSE
RT THE PROGRAM DISKETTE ":? "
HIT RETURN WHEN READY":CLOSE #2
1010 OPEN #2,4,0,"K:":GET #2,X:TRAP 10
20:OPEN #1,4,0,"D:TRACK.BAS":CLOSE #1:
CLOSE #2:RUN "D:TRACK.BAS"
1020 TRAP 40000:CLOSE #1:? "■■":GOTO 1
000
```

# KOOKY DIVER

```
100 GOTO 620
110 OX=100:SOUND 1,200,8,1:SOUND 2,201
,10,1:SOUND 3,202,10,2
120 OX=OX-0.25-TRT/32*TR:POSITION 10,2
3:? INT(OX)*(OX>0);" ";:POSITION 21,23
:? SC;:POSITION 37,23:? LI;
130 IF OX<1 THEN 930
140 IF CU AND RND(C0)>0.95-TRT/800 THE
N 350
150 IF  NOT RA AND CD AND RND(C0)>0.95
 THEN POSITION 20,18:? "!&↓←←+*↓←←+*↓←
←#@(((()":CU=1:CD=C0:POKE PLX+1,139
160 ST=STICK(0):IF ST=15 THEN POKE PDR
,1:PY=PY+1*(PY<205)
170 IF ST<>15 THEN POKE PDR,INT(RND(C0
)*2+1)*12+1:POKE 77,0:IF OX<50 THEN OL
=1
180 IF ST=14 THEN PY=PY-3*(PY>49)
190 IF ST=13 THEN PY=PY+3*(PY<205)
200 IF ST=11 THEN YX=YX-3*(YX>49):PY=P
Y+1*(PY<205)
210 IF ST=7 THEN YX=YX+3*(YX<200):PY=P
Y+1*(PY<205)
220 SRP=SRP-1:POKE DL+7,SRP:POKE DL+10
,SRP:IF SRP=0 THEN SRP=40
230 IF PY<52 AND OL THEN OX=100:OL=C0:
FOR X=240 TO 250:SOUND 0,X,2,6:NEXT X:
SOUND C0,C0,C0,C0
240 POKE PLY,PY:POKE PLX,YX:TFB=PEEK(5
3252):IF PEEK(53260)=0 AND TFB<>3 AND
TFB<>5 AND TFB<>7 THEN 120
250 IF PEEK(53260)<>2 AND PEEK(53260)<
>0 THEN 930
260 IF PEEK(53252)=3 AND PY>70 AND TR
THEN 330
270 IF PEEK(53260) AND STRIG(0)=0 THEN
 TR=1:RA=1:POKE PLX+1,0:FOR W=1 TO 15:
SOUND 0,50,10,8:NEXT W:SOUND 0,0,0,0
280 IF PEEK(53252)=3 AND PY<60 AND TR
THEN POKE PLY+1,PY-10:POKE PLX+1,YX:GO
TO 310
290 IF (CD AND PEEK(53252)=7 AND PY>19
0) OR (CD AND PEEK(53252)=5 AND YX>100
 AND YX<160 AND PY>100) THEN 930
300 POKE 53278,0:GOTO 120
310 FOR JW=200 TO 210:SOUND 0,JW,10,8:
NEXT JW:SOUND C0,C0,C0,C0:FOR MIAD=1 T
O 150:NEXT MIAD
320 SC=SC+TRT*100:TR=C0:RA=C0:POKE PLX
+1,0:POKE PLY+1,198:TRT=INT(RND(C0)*8)
*8+1:POKE PDR+1,TRT:GOTO 300
330 POKE PLY+1,161*(PY>150)+113*(PY>10
0 AND PY<140)+81*(PY>75 AND PY<90):TR=
0:FOR GH=220 TO 225:SOUND 0,GH,10,8
340 NEXT GH:SOUND C0,C0,C0,C0:POKE PLX
+1,YX:GOTO 270
350 FOR Y=18 TO 21:POSITION 20,Y:? "CC
":NEXT Y:POSITION 21,21:? "[;;;]":CD=1
:CU=C0:POKE PLX+1,0:GOTO 150
360 DIM CH$(40):CH$="ABCDEFGHIJKLMNOPQ
RSTUVWXYZ!+#$%&*@()[]=;":PM=(PEEK(106)
-16)*256:CH=(PEEK(106)-8)*256
370 FOR I=0 TO 511:POKE CH+I,PEEK(5734
4+I):NEXT I:FOR I=1 TO 40:CHP=CH+(ASC(
CH$(I))-32)*8:FOR J=0 TO 7:READ A
380 POKE CHP+J,A:NEXT J:NEXT I:POKE 75
6,CH/256:POKE 752,1:M=PEEK(106)-14:N=M
+1
390 DL=PEEK(560)+256*PEEK(561):POKE DL
+3,68:FOR X=DL+6 TO DL+32:POKE X,4:NEX
T X:POKE 82,0:SM=PEEK(88)+256*PEEK(89)

400 POKE DL+7,0:POKE DL+8,M:POKE DL+13
,PEEK(88)+120:POKE DL+14,PEEK(89):POKE
 DL+33,132:POKE DL+35,65
410 POKE DL+4,68:POKE DL+6,68:POKE DL+
12,68:POKE DL+9,68:POKE DL+10,0:POKE D
L+11,N
420 OX=100:LI=3:FOR X=1710 TO 1769:REA
D A:POKE X,A:NEXT X
430 POKE 88,0:POKE 89,M:POSITION 5,0:?
 "F==========E":POSITION 5,1:? "F=====
=====E"
440 POKE 89,N:POSITION 0,0:? "ABAABBGD
DDDDDDDHBABAABABBAABAABABAABBAAB"
450 POSITION 0,1:? "ABAABBGDDDDDDDDDHBA
BAABABBAABAABABAABBAAB"
460 POKE 89,INT(SM/256):POKE 88,SM-PEE
K(89)*256
470 POSITION 35,7:? "JDDDD":POSITION 3
5,8:? "KIIII":POSITION 36,9:? "LMNO"
480 POSITION 0,11:? "DDDDDDT":? "IIIII
IU":? "IIIIIV";? "PQRRS":POSITION 31,1
7:? "JDDDDDDDD"
490 POSITION 31,18:? "KIIIIIIII":POSIT
ION 31,19:? "WIIIIIIII":POSITION 31,20
:? "XYZPQRRS"
500 POSITION 20,18:? "!&":POSITION 20,
19:? "+*":POSITION 20,20:? "+*":POSITI
ON 20,21:? "#@(((()"
510 POSITION 21,22:? "#$$$%":FOR X=SM+
120 TO SM+919:IF PEEK(X)=0 THEN POKE X
,35
520 NEXT X:FOR X=704 TO 710:READ A:POK
E X,A:NEXT X
530 FOR I=1536 TO 1706:READ A:POKE I,A
:NEXT I:FOR I=1774 TO 1787:POKE I,0:NE
XT I:FOR I=PM+1023 TO PM+2047:POKE I,0

540 NEXT I:FOR I=PM+1536 TO PM+1543:RE
AD A:POKE I,A:NEXT I:FOR I=PM+1792 TO
PM+1799:READ A:POKE I,A:NEXT I
550 FOR I=PM+1 TO PM+48:READ A:POKE I,
A:NEXT I
560 FOR I=PM+257 TO PM+320:READ A:POKE
 I,A:NEXT I:PLX=53248:PLY=1780:PLL=178
4:PDR=1772:POKE 623,1
570 POKE 1788,PM/256+4:POKE 53277,3:PO
KE 54279,PM/256:POKE 1771,PM/256:X=USR
(1696)
580 POKE PLL,12:POKE PLL+1,8:POKE PLL+
2,8:POKE PLL+3,8:POKE 53258,3:POKE 532
59,3
590 POSITION 2,23:? "OXYGEN: 99  SCORE
: 0       DIVERS: 3";
600 POKE 208,0:POKE DL+34,130:POKE 512
,174:POKE 513,6:POKE 54286,192:SRP=40:
CU=1:C0=0
610 YX=90:PY=49:POKE PLX+1,139:POKE PL
Y+1,198:POKE PDR+1,1:POKE PLY+2,155:PO
KE PLY+3,65:TRT=1:POKE 559,62:GOTO 110

620 GRAPHICS 1:POKE 16,64:POKE 53774,6
4:SETCOLOR 2,0,0:DL=PEEK(560)+256*PEEK
(561):POKE DL+14,23:POKE DL+18,2
630 POKE 54276,4:SETCOLOR 0,8,4:POSITI
ON 6,9:? #6;"KOOKY DIVER":POSITION 4,1
1:? #6;"press start to begin"
640 POSITION 9,13:? #6;"ANSCHUETZ/WEIS
GERBER/ANSCHUETZ"
650 IF PEEK(53279)=6 THEN GRAPHICS 0:P
OKE 16,64:POKE 53774,64:POKE 559,0:POK
E 712,134:GOTO 360
660 GOTO 650
670 DATA 0,0,0,15,55,220,215,215,0,0,0
,0,0,0,0,255,85,85,85,85,85,85,85,1
70,170,170,170,170,170,170,170,170
680 DATA 170,170,168,168,160,160,128,1
28,170,170,42,42,10,10,2,2,170,170,42,
42,10,90,66,254
690 DATA 170,170,168,168,160,172,151,1
51,255,255,255,255,255,255,255,255,87,
95,95,127,255,127,255,127
700 DATA 127,95,127,255,255,95,127,87,
255,127,95,127,127,255,127,87,255,255,
255,255,253,253,245,213
710 DATA 255,253,213,85,85,85,85,85,25
5,127,95,127,95,255,95,87,255,125,85,8
5,85,85,85,85
720 DATA 255,127,95,87,85,85,85,85,255
,255,255,255,221,85,85,85,255,253,213,
85,85,85,85,85
730 DATA 213,245,245,213,245,245,213,2
13,213,213,245,253,245,253,245,213,255
,255,253,255,253,253,245,213
740 DATA 87,87,95,127,127,95,87,87,255
,127,95,95,127,125,117,85,255,255,247,
247,213,85,85,85
750 DATA 255,255,253,253,117,85,85,85,
87,87,94,94,122,122,234,234,234,234,23
4,234,234,234,234,234
760 DATA 234,234,122,122,94,94,87,87,1
70,170,170,170,170,170,255,255,171,171
,173,173,181,181,213,213
770 DATA 213,213,245,245,221,221,221,2
21,221,221,221,221,221,221,221,221,221
,221,223,223,245,245,213,255
780 DATA 85,85,255,255,85,85,85,255,85
,85,85,213,245,125,95,255,87,87,94,94,
122,122,234,255
790 DATA 213,213,181,181,173,173,171,2
55,170,170,255,255,255,255,170,170,255
,255,170,170,170,170,170,255
800 DATA 72,169,224,141,10,212,141,9,2
12,169,212,141,24,208,169,195,141,0,2,
104,64
810 DATA 72,169,250,197,208,208,7,169,
0,133,208,24,144,15,230,208,165,208,14
1,2,208,169,255,56,229,208
820 DATA 141,3,208,141,10,212,169,174,
141,0,2,104,64,246,0,128,6,132,216,50
830 DATA 162,3,189,244,6,240,89,56,221
,240,6,240,83,141,254,6,106,141,255,6,
142,253,6,24,169,0,109,253,6,24,109
840 DATA 252,6,133,204,133,206,189,240
,6,133,203,173,254,6,133,205,189,248,6
,170,232,46,255,6,144,16,168,177,203
850 DATA 145,205,169,0,145,203,136,202
,208,244,76,87,6,160,0,177,203,145,205
,169,0,145,203,200,202,208,244,174
```

```
860 DATA 253,6,173,254,6,157,240,6,189
,236,6,240,48,133,203,24,138,141,253,6
,109,235,6,133,204,24,173,253,6,109
870 DATA 252,6,133,206,189,240,6,133,2
05,189,248,6,170,160,0,177,203,145,205
,200,202,208,248,174,253,6,169,0,157
880 DATA 236,6,202,48,3,76,2,6,76,98,2
28,0,0,104,169,7,162,6,160,0,32,92,228
,96,8,156,190,251,254,191,156,8,8
890 DATA 65,16,130,16,69,0,146,16,56,5
6,146,254,56,56,56,40,40,40,108,16,58,
57,17,255,184,184,190
900 DATA 34,35,32,96,8,92,156,136,255,
29,29,125,68,196,4,6,108,40,40,40,56,5
6,56,254,146,56,56,16
910 DATA 0,0,0,24,60,118,60,24,16,56,8
4,124,40,56,16,254,0,126,24,36,66,66,3
6,24,40,42,252,39,228,63,84,20
920 DATA 0,0,159,104,232,208,192,192,1
,2,12,28,184,112,96,144,62,28,127,93,6
2,28,8,28,0,0,254,254,65,127,119,127
930 POKE PDR,37:FOR MB=PY TO 205:POKE
PLY,MB:SOUND 0,MB,10,8:NEXT MB:SOUND C
0,C0,C0,C0:FOR W=1 TO 25:NEXT W
940 IF TR THEN TR=0:RA=0:POKE PLX+1,0:
POKE PLY+1,198
950 LI=LI-1:YX=90:PY=49:OL=C0:IF LI=0
THEN POSITION 37,23:? "0";:GOTO 970
960 POKE 559,0:POKE PDR,1:POKE PLY,PY:
POKE PLX,YX:POKE 53278,0:POKE 559,62:G
OTO 110
970 FOR FIAB=1 TO 3:SOUND FIAB,C0,C0,C
0:NEXT FIAB:CU=1:CD=0:PX=0:TR=C0:RA=C0

980 GOSUB 1050:GOSUB 1060:IF SC>HS THE
N HS=SC
990 POSITION 0,23:? "HIGH SCORE: ";HS;
:POSITION 19,23:? "PRESS START TO BEGI
N";:FOR W=1 TO 50:GOSUB 1060:NEXT W
1000 GOSUB 1050:POSITION 2,23:? "OXYGE
N: ";INT(OX)*(OX>0);:POSITION 14,23:?
"SCORE: ";SC;:POSITION 29,23
1010 ? "DIVERS: ";LI;:FOR W=1 TO 50:GO
SUB 1060:NEXT W:GOTO 970
1020 GOSUB 1050
1030 POSITION 20,18:POKE 559,0:? "!&↓←
←+*↓←←+*↓←←#@(((()":POSITION 2,23:? "OX
YGEN: 99  SCORE: 0       DIVERS: 3";
1040 POKE PDR,1:POKE PLX,90:POKE PLY,4
9:POKE 53278,0:LI=3:SC=C0:GOTO 610
1050 POSITION 0,23:? "                ";:RETURN
1060 IF PEEK(53279)=6 THEN POP :GOTO 1
020
1070 RETURN
```

# KOOKY KLIMBER

```
100 GOTO 330
110 POKE PLX+1,X5:IF PEEK(53253)>=4 AN
D PEEK(53253)<=7 THEN POKE PLX+1,0:POK
E 53278,0:GOTO FALL
120 POKE PLX+1,0
130 QR=-QR:IF QR=1 THEN GOSUB 660
140 GOSUB 710:POKE PDR,1:IF INT(WC*RND
(0)+1)=1 THEN 240
150 R=INT(5*RND(0)+1)*40+ML-70:R1=R+IN
T(7*RND(0)+1)
160 L=170:T=0:GOTO 190
170 L=180:T=10:GOTO 190
180 L=110:T=20
190 P=PEEK(R1+T)
200 IF P=90 THEN POKE R1+T,95:POKE R1+
T+1,245:GOTO 110
210 IF P=170 THEN POKE R1+T,255:GOTO 1
10
220 IF P=165 THEN POKE R1+T,245:POKE R
1+T-1,95:GOTO 110
230 GOTO L
240 R=INT(5*RND(0)+1)*40+ML-50:R1=R+IN
T(7*RND(0)+1)
250 L=260:T=0:GOTO 280
260 L=270:T=10:GOTO 280
270 L=110:T=20
280 P=PEEK(R1-T)
290 IF P=95 THEN POKE R1-T,90:POKE R1-
T+1,165:GOTO 110
300 IF P=255 THEN POKE R1-T,170:GOTO 1
10
310 IF P=245 THEN POKE R1-T,165:POKE R
1-T-1,90:GOTO 110
320 GOTO L
330 GOSUB 1180:PM=PEEK(106)-16:PMBASE=
256*PM:DL=PMBASE+2083:ST=1:IL=3:VX=1:P
COL0=230:PCOL1=230:PCOL2=196:PCOL3=66
340 GOSUB 370:POKE PLL,34:POKE PLL+1,2
:POKE PLL+2,16:POKE PLL+3,16:POKE 5325
6,1:POKE 53257,1:POKE 53258,1
350 X5=120:POKE PLY,137:POKE PLY+1,137
:Y8=226:JW=1:QR=-1:FALL=830
360 DRAW=1:DRAW1=1:POKE PDR,DRAW:POKE
PDR+2,DRAW1:MLB=INT(DL/256):MLA=DL-MLB
*256:GOTO 650
370 FOR I=1536 TO 1706:READ A:POKE I,A
:NEXT I:FOR I=1774 TO 1787:POKE I,0:NE
XT I
380 FOR I=PMBASE+1023 TO PMBASE+2047:P
OKE I,0:NEXT I:DRWBAS=PMBASE+1
390 FOR K=DRWBAS TO DRWBAS+135:READ X:
POKE K,X:NEXT K:POKE PMBASE+1280,231:P
OKE PMBASE+1281,231
400 FOR K=PMBASE+513 TO PMBASE+576:REA
D X:POKE K,X:NEXT K
410 FOR K=PMBASE+1792 TO PMBASE+1807:R
EAD X:POKE K,X:NEXT K
420 POKE 704,PCOL0:POKE 705,PCOL1:POKE
 706,PCOL2:POKE 707,PCOL3
430 PLX=53248:PLY=1780:PLL=1784:POKE 5
59,62:POKE 623,1:POKE 1788,PM+4:POKE 5
3277,3:POKE 54279,PM
440 PDR=1772:POKE 1771,PM:X=USR(1696):
RETURN
450 DATA 162,3,189,244,6,240,89,56,221
,240,6,240,83,141,254,6,106,141
460 DATA 255,6,142,253,6,24,169,0,109,
253,6,24,109,252,6,133,204,133
470 DATA 206,189,240,6,133,203,173,254
,6,133,205,189,248,6,170,232,46,255
480 DATA 6,144,16,168,177,203,145,205,
169,0,145,203,136,202,208,244,76,87
490 DATA 6,160,0,177,203,145,205,169,0
,145,203,200,202,208,244,174,253,6
500 DATA 173,254,6,157,240,6,189,236,6
,240,48,133,203,24,138,141,253,6
510 DATA 109,235,6,133,204,24,173,253,
6,109,252,6,133,206,189,240,6,133
520 DATA 205,189,248,6,170,160,0,177,2
03,145,205,200,202,208,248,174,253,6
530 DATA 169,0,157,236,6,202,48,3,76,2
,6,76,98,228,0,0,104,169
540 DATA 7,162,6,160,0,32,92,228,96
550 DATA 231,231,195,195,66,66,90,90,9
0,126,126,126,60,60,60,60,60,24,24,24,
24,24,24,24,60,60,60,36,36,36,102
560 DATA 102,231,231
570 DATA 224,224,192,192,71,71,67,67,9
0,90,90,126,126,126,60,60,60,60,60,24,
24,24,24,24,60,60,60,100,100,228,228
580 DATA 6,6,7,7,7,7,3,3
590 DATA 226,226,194,194,90,90,90,126,
126,126,60,60,60,60,60,24,24,24,24,60,
60,60,38,38,39,39,96,96,224,224
600 DATA 231,231,102,102,36,36,36,60,6
0,60,24,24,24,24,24,24,24,60,60,60,60,
60,126,126,126
610 DATA 90,90,90,66,66,195,195,231,23
1
620 DATA 0,0,129,129,129,129,255,255,1
29,129,129,129,0,0,0,0,16,16,32,32,96,
96,144,144,9,9,6,6,4,4,8,8
630 DATA 124,124,16,16,16,16,16,16,16,
16,16,16,16,16,124,124,8,8,4,4,6,6,9,9
,144,144,96,96,32,32,16,16
640 DATA 64,64,224,226,84,72,4,4,255,2
55,255,126,126,126,60,60
650 GOSUB 1050:GOTO 110
660 IF (STICK(0)=14 OR STICK(0)=6 OR S
TICK(0)=10) AND ST=1 THEN ST=0:POKE 77
,0:GOSUB 920
670 IF STICK(0)=13 OR STICK(0)=5 OR ST
ICK(0)=9 THEN ST=1
680 IF STICK(0)=11 THEN X5=X5-12:GOSUB
 1230:IF X5=48 THEN GOTO FALL
690 IF STICK(0)=7 THEN X5=X5+12:GOSUB
1230:IF X5=192 THEN GOTO FALL
700 RETURN
710 IF Y8>225 OR Y9>225 THEN Y8=0:Y9=0
:X8=0:X9=0:POKE PLX+2,0:POKE PLY+2,0:P
OKE PLX+3,0:POKE PLY+3,0:GOSUB 760
720 IF EA=1 AND JW THEN POKE PDR+2,DRA
W1:Y8=Y8+6+RA:DRAW1=DRAW1+16:IF DRAW1>
50 THEN DRAW1=1
730 IF EA=2 AND JW THEN Y9=Y9+3+RA
740 IF STICK(0)=13 OR STICK(0)=5 OR ST
ICK(0)=9 THEN ST=1
750 GOTO 800
760 EA=INT(RND(0)*2)+1
770 IF EA=1 AND JW THEN X8=INT(RND(0)*
120+60):Y8=10:RETURN
780 IF JW THEN X9=INT(RND(0)*120+60):Y
9=10:DRAW1=1:RETURN
790 RETURN
800 POKE PLX+2,X8:POKE PLY+2,Y8:POKE P
LX+3,X9:POKE PLY+3,Y9
810 IF PEEK(53262) OR PEEK(53263) THEN
 POKE 53278,0:GOTO FALL
820 POKE PLX,X5:ML=PE+256*PE1:RETURN
830 ML1=ML:POKE PDR,103:POKE PLY+2,0:P
OKE PLX+2,0:POKE PLY+3,0:POKE PLX+3,0:
Y8=226:Y9=226
840 IL=IL-1:IF IL=0 THEN 1240
850 IF ML=DL+1720 THEN POKE PLX,0:X5=0
:POKE 53278,0:ST=1:GOTO 880
860 PE=PE+10:IF PE>255 THEN PE=PE-256:
PE1=PE1+1:IF PE1>255 THEN PE1=PE1-256
870 ML=PE+256*PE1:POKE DL+4,PE:POKE DL
+5,PE1:GOTO 850
880 FOR X=ML1+1 TO DL+1881 STEP 10:IF
PEEK(X)=85 THEN NEXT X:GOTO 1260
890 FOR X1=X TO X+6:IF PEEK(X1)=95 THE
N POKE X1,90:X1=X1+1:POKE X1,165:NEXT
X1:NEXT X:GOTO 1260
900 IF PEEK(X1)=255 THEN POKE X1,170:N
EXT X1
910 NEXT X1:NEXT X:GOTO 1260
920 FOR X=1 TO 4:PE=PE-10:IF PE<0 THEN
 PE=PE+256:PE1=PE1-1:IF PE1<0 THEN PE1
=PE1+256
930 SC=SC+PT:POKE DL+4,PE:POKE DL+5,PE
1
940 IF SC>=10000 AND VX THEN IL=IL+1:V
X=0:FOR W=10 TO 30:SOUND 1,W,10,10:NEX
T W:SOUND 1,0,0,0
950 IF X=1 OR X=3 THEN DRAW=35
960 IF X=2 THEN DRAW=69
970 IF X=4 THEN DRAW=1
980 POKE PDR,DRAW:ML=PE+256*PE1
990 IF EA=1 AND JW THEN POKE PDR+2,DRA
W1:Y8=Y8+14+RA:DRAW1=DRAW1+16:POKE PLY
+2,Y8*(Y8<255)
1000 IF EA=2 AND JW THEN Y9=Y9+11+RA:P
OKE PLY+3,Y9*(Y9<255)
1010 GOSUB 1230:IF DRAW1>50 THEN DRAW1
=1
1020 IF DL+205>ML THEN JW=1:GOSUB 1050
:GOTO 110
1030 IF ML<DL+445 THEN JW=0:POKE PLX+2
,0:POKE PLY+2,0:POKE PLX+3,0:POKE PLY+
3,0:Y8=226:Y9=226
1040 NEXT X:RETURN
1050 POKE PLX+2,0:POKE PLY+2,0:POKE PL
X+3,0:POKE PLY+3,0:Y8=226:Y9=226:RA=RA
+1:PT=RA*5:IF PT>50 THEN PT=50
1060 POKE PLX,0:X5=120:WC=RA*2+1:SC=SC
+(RA-1)*1000
1070 GRAPHICS 17:SETCOLOR 4,8,6:SETCOL
OR 0,13,2
1080 BN=BN+1:POSITION 4,9:? #6;"SCORE:
 ";SC:POSITION 4,11:? #6;"BUILDING #";
BN:POSITION 4,13:? #6;"KLIMBERS: ";IL
1090 FOR W=1 TO 500:NEXT W:GRAPHICS 21
:SETCOLOR 0,(RA-1)*2,4:SETCOLOR 1,0,0:
SETCOLOR 2,14,10:SETCOLOR 4,8,6
1100 ST=1:POKE 560,MLA:POKE 561,MLB:PO
KE 559,0:POKE DL,0:POKE DL+1,0:POKE DL
+2,0:POKE DL+35,65
1110 POKE DL+36,MLA:POKE DL+37,MLB:POK
E DL+3,72:FOR X=DL+6 TO DL+34:POKE X,8
```

```
:NEXT X
1120 FOR X=DL+830 TO DL+910:POKE X,0:N
EXT X:FOR X=DL+401 TO DL+1881 STEP 40:
FOR X1=X TO X+7:POKE X1,85:NEXT X1
1130 FOR X1=X+10 TO X+30 STEP 10:FOR X
2=X1 TO X1+6 STEP 3:POKE X2,90:POKE X2
+1,165:IF X2<X1+6 THEN POKE X2+2,170
1140 NEXT X2:NEXT X1:NEXT X:FOR X=DL+4
00 TO DL+1881 STEP 200:POKE X,1:POKE X
+9,64:NEXT X
1150 FOR X=DL+1921 TO DL+1928:POKE X,8
5:NEXT X:FOR X=DL+1930 TO DL+1979:POKE
 X,85:NEXT X
1160 X=INT((DL+1720)/256):POKE DL+4,DL
+1720-X*256:POKE DL+5,X:PE=DL+1720-X*2
56:PE1=X
1170 ML=PEEK(DL+4)+256*PEEK(DL+5):POKE
 559,62:POKE PLX,X5:RETURN
1180 GRAPHICS 1:SETCOLOR 2,0,0:DL=PEEK
(560)+256*PEEK(561):POKE DL+14,23:POKE
 DL+18,2:POKE 54276,4
1190 SETCOLOR 0,8,4:POSITION 5,9:? #6;
"KOOKY KLIMBER":POSITION 4,11:? #6;"pr
ess start to begin":POSITION 9,13
1200 ? #6;"ANSCHUETZ/WEISGERBER/ANSCHU
ETZ"
1210 IF PEEK(53279)=6 THEN GRAPHICS 21
:SETCOLOR 4,8,6:RETURN
1220 GOTO 1210
1230 POKE PLX,X5:FOR W=0 TO 10 STEP 1.
5:SOUND 0,220-W,10,10-W:SOUND 0,180-W,
10,10-W:NEXT W:SOUND 0,0,0,0:RETURN
1240 FOR X=0 TO 3:POKE PLX+X,0:NEXT X:
POKE 53277,0:GRAPHICS 17:SETCOLOR 2,0,
0
1250 POSITION 4,9:? #6;"SCORE ";SC:POS
ITION 5,16:? #6;"game over":FOR W=0 TO
 0 STEP 0:NEXT W
1260 POKE PDR,1:POKE PLX,120:X5=120:GO
TO 110
```

# KOOKY'S QUEST

```
10 REM KOOKY'S QUEST
20 REM BY
30 REM ANSCHUETZ/WEISGERBER/ANSCHUETZ
40 REM ANTIC MAGAZINE
100 GRAPHICS 18:POKE 16,64:POKE 53774,
64:DL=PEEK(560)+256*PEEK(561):SETCOLOR
 2,0,0
110 POKE DL+10,6:POKE DL+11,6:POKE DL+
12,6:POKE DL+13,2:SETCOLOR 0,8,4:POSIT
ION 4,4
120 ? #6;"KOOKY'S QUEST":POSITION 5,8:
? #6;"ANSCHUETZ/WEISGERBER/ANSCHUETZ"
130 POSITION 0,6:? #6;"press start to
begin":IF PEEK(53279)<>6 THEN 130
140 DIM A$(25),B$(25):GRAPHICS 0:POKE
16,64:POKE 53774,64:POKE 752,1:SETCOLO
R 2,0,2:SETCOLOR 4,0,2
150 A$="↓↓↓↓↓↓ ↓↓ ↓↓ ↓↓↓↓↓↓↓":B$=
" ↓↓ ↓↓ ↓↓ ↓↓ ↓↓ ↓↓ ↓↓ ↓↓ "
160 DIM OHP(6,6),MON$(6),CON$(6),T$(2)
,WEP$(12):MON$="^;(/:\":CON$="[     ":
T$="]_":WEP$="ghefcdabijkl"
170 FOR R=1 TO 6:FOR C=1 TO 6:READ A:O
HP(R,C)=A:NEXT C:NEXT R
180 DIM F$(16):F$="↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓"
190 DIM G$(25),H$(25):G$="↓↓↓↓↓↓↓↓↓↓
↓↓↓↓↓↓↓↓↓↓↓↓↓":H$="↓↓↓ ↓↓ ↓↓ ↓↓ ↓↓ ↓↓
↓↓ ↓↓↓"
200 DATA 30,15,10,6,5,7.5,10,30,15,5,7
.5,6,15,10,30,7.5,6,5,5,7.5,6,15,30,10
,7.5,6,5,10,15,30,6,5,7.5,30,10,15
210 DATA 0,0,64,248,64,0,0,0
220 DATA 0,0,2,31,2,0,0,0
230 DATA 0,27,10,254,10,27,0,0
240 DATA 0,216,80,127,80,216,0,0
250 DATA 0,31,14,254,14,31,0,0
260 DATA 0,248,112,127,112,248,0,0
270 DATA 0,64,96,255,96,64,0,0
280 DATA 0,2,6,255,6,2,0,0
290 DATA 0,0,63,24,252,0,0,0
300 DATA 0,0,252,24,63,0,0,0
310 DATA 0,40,18,92,58,72,20,0
320 DATA 0,20,72,58,92,18,40,0
330 DATA 255,135,219,219,219,219,135,2
55
340 DATA 126,255,255,231,231,255,255,1
26
350 DATA 60,24,36,66,66,36,24,0
360 DATA 85,85,85,85,85,85,85,85
370 DATA 254,254,254,0,239,239,239,0
380 DATA 0,126,219,255,189,195,126,0
390 DATA 255,231,195,195,231,231,231,2
55
400 DATA 255,219,147,219,219,219,193,2
55
410 DATA 126,255,255,255,255,255,255,1
26
420 DATA 73,114,149,12,126,146,34,85
430 DATA 28,62,42,28,42,34,85,85
440 DATA 36,219,126,66,255,165,36,231
450 DATA 0,0,219,126,24,36,0,0
460 DATA 240,80,248,30,7,33,67,62
470 DATA 0,28,42,28,73,62,8,54
480 DATA 124,84,124,40,254,56,124,238
490 DATA 0,0,192,224,191,234,192,0
500 DATA 60,66,129,255,165,165,255,0
510 DATA 170,170,170,170,170,170,170,1
70
520 ? "K"
530 DIM CHA$(31):CHA$="abcdefghijkl*+_
@<>%&#[^;:/\()]?":CHS=(PEEK(106)-8)*25
6
540 PM=(PEEK(106)-12)*256:IF PEEK(CHS+
9)<>0 THEN 590
545 ? :? :? "    One moment please..."
:FOR X=1 TO 700:NEXT X:? "K":POKE 559,
0
550 FOR I=0 TO 1023:POKE CHS+I,PEEK(57
344+I):NEXT I:FOR I=1 TO 31:CHP=CHS+(A
SC(CHA$(I))-32)*8
560 IF I<13 THEN CHP=CHP+256
570 FOR J=0 TO 7:READ A:POKE CHP+J,A:N
EXT J:NEXT I
580 FOR X=PM+64 TO PM+128:POKE X,255:N
EXT X
590 POKE 53277,3:POKE 54279,PM/256-4:P
OKE 623,1:POKE 53256,3:POSITION 2,12:P
OKE 559,62
600 OPEN #1,4,0,"K:":? "→→→→SKILL - EX
PERT OR BEGINNER";:GET #1,K:IF K<>69 A
ND K<>66 THEN CLOSE #1:? "↑":GOTO 600
610 ? "K":POKE 559,0:POKE 704,2:P=188:
SC=PEEK(88)+256*PEEK(89):L=1:X=30:X1=S
C+472:LI=10:RL=7
620 MO1=4:MO2=12:Q=1:ST=23:TS=15:POKE
82,0
630 POKE 756,CHS/256:GOSUB 1870
640 POSITION 4,3:? "KOOKY'S↓↓↓↓↓↓↓QUES
T":POSITION 1,6:? "SCORE=0↓↓↓↓↓↓↓↓↓LIV
ES=10↓↓↓↓↓↓↓↓↓↓LEVEL=1"
650 POSITION 1,12:? "TREASURES=0"
660 FOR Y=18 TO 21:POSITION 0,Y:? "<<<
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<"
;:NEXT Y
670 POSITION 27,3:? "??????????":POSIT
ION 27,12:? "??????????"
680 FOR R=4 TO 11:POSITION 27,R:? "?":
POSITION 36,R:? "?":NEXT R
690 POSITION 5,19:? "1 SWORD   3 MACE
   5 L.BOLT  ":POSITION 5,20:? "2 AXE
   4 KNIFE   6 FIREBALL"
700 POSITION 14,3:? "@@@@@@@@@@@":POSI
TION 14,13:? "@@@@@@@@@@@":FOR R=4 TO
12:POSITION 14,R:? "@":POSITION 24,R
710 ? "@":NEXT R:FOR R=4 TO 6:POSITION
 15,R:? "<<<    <<<":POSITION 15,R+6:?
"<<<<<<<<<":NEXT R
720 N=PEEK(X1):POKE X1,N+128:POSITION
2,0:? " ":POKE 704,42*(K=69)+132*(K=66
):POKE 559,62:POKE 53248,160
730 J=STICK(0)
740 IF  NOT (CR=RL AND ((DMT<5 AND L>1
) OR (DMT<1 AND L=1))) THEN 760
750 POSITION 4,16:? "YOU WILL WANDER L
EVEL ";L;" FOREVER":FOR MIAD=0 TO 0 ST
EP 0:NEXT MIAD
760 IF J=14 AND (N=65 OR N=67 OR N=68
OR N=88 OR N=90 OR N=83 OR N=124 OR N<
11) THEN ST=1:A=7:A1=9:GOTO 820
770 IF J=13 AND (N=65 OR N=81 OR N=68
OR N=69 OR N=87 OR N=83 OR N=124 OR N<
11) THEN ST=-1:A=9:A1=7:GOTO 820
780 IF J=7 AND (N=65 OR N=81 OR N=87 O
R N=82 OR N=88 OR N=90 OR N=83) THEN S
T=-1:A=20:A1=18:GOTO 1020
790 IF J=11 AND (N=67 OR N=68 OR N=69
OR N=87 OR N=82 OR N=88 OR N=83) THEN
ST=1:A=18:A1=20:GOTO 1020
800 IF N=6 OR N=10 THEN 2060
810 GOTO 730
820 V=ST*3:POKE X1,N
830 IF N=124 OR N=6 OR N=10 THEN 850
840 GOSUB 950*(N=69 OR N=68 OR N=67)+9
80*(N=65 OR N=90 OR N=81)+1000*(N=83 O
R N=87 OR N=88)
850 A=A-V:A1=A1-V:X1=X1-ST*40:N=PEEK(X
1):POKE X1,N+128
860 IF N=65 THEN 1890
870 IF N=68 THEN 1910
880 IF N=67 OR N=69 THEN 1930
890 IF N=87 OR N=88 THEN 1950
900 IF N=81 OR N=90 THEN 1970
910 IF N=83 THEN 1990
920 IF N=3 OR N=11 THEN 2010
930 IF N=6 OR N=10 THEN 2060
940 POKE 77,0:GOTO 730
950 FOR W=A TO A1 STEP ST:POSITION 15,
W:? "<<<":POSITION 15,W+V:? "      ":N
EXT W
960 FOR W=A TO A1 STEP ST:POSITION 15,
W+V:? "<<<":NEXT W:RETURN
970 GOTO 730
980 FOR W=A TO A1 STEP ST:POSITION 21,
W:? "<<<":POSITION 18,W+V:? "      ":N
EXT W
990 FOR W=A TO A1 STEP ST:POSITION 21,
W+V:? "<<<":NEXT W:RETURN
1000 FOR W=A TO A1 STEP ST:POSITION 15
,W:? "<<<   <<<":POSITION 15,W+V:? "
     ":NEXT W
1010 FOR W=A TO A1 STEP ST:POSITION 15
,W+V:? "<<<   <<<":NEXT W:RETURN
1020 V=ST*3:POKE X1,N
1030 DQ=2050*(N=65 OR N=68 OR N=83)+21
00*(N=67 OR N=88 OR N=90)+2150*(N=69 O
R N=87 OR N=81)
1040 IF N<>82 THEN GOSUB 1140*(N=65 OR
 N=68 OR N=83)+1160*(N=67 OR N=88 OR N
=90)+1180*(N=69 OR N=87 OR N=81)
1050 A=A-V:A1=A1-V:X1=X1-ST:N=PEEK(X1)
:POKE X1,N+128
1060 IF N=88 THEN 2130
1070 IF N=83 THEN 2150
1080 IF N=90 OR N=67 THEN 2170
1090 IF N=65 OR N=68 THEN 2190
1100 IF N=69 OR N=81 THEN 2210
1110 IF N=87 THEN 2230
1120 IF N=3 OR N=11 THEN 2250
1130 POKE 77,0:GOTO 730
1140 FOR W=A TO A1 STEP ST:POSITION W+
V,4:? B$:POSITION W,4:? A$:NEXT W
1150 FOR W=A TO A1 STEP ST:POSITION W+
V,4:? A$:NEXT W:RETURN
1160 FOR W=A TO A1 STEP ST:POSITION W+
V,4:? B$(10):POSITION W,4:? G$(19):NEX
T W
1170 FOR W=A TO A1 STEP ST:POSITION W+
V,4:? G$(19):NEXT W:RETURN
1180 FOR W=A TO A1 STEP ST:POSITION W+
V,7:? B$(10):POSITION W,10:? G$(19):NE
```

```
XT W
1180 FOR W=A TO A1 STEP ST:POSITION W+
V,10:? G$(19):NEXT W:RETURN
1200 QP=0:DM=0:GOTO 1210
1210 Z=INT(RND(0)*2):M3=16+6*Z:MM3=8:D
M=0:MON=INT(RND(1)*5)+2:KX=19:KY=8:THP
=0:IF L=5 THEN MON=1
1220 TX=INT(RND(0)*7)+16:TY=INT(RND(0)
*7)+5:TY=TY+(TY=8):POSITION TX,TY:? T$
(Z+1,Z+1)
1230 IF L=1 AND X1=SC+428 THEN POSITIO
N 19,5:? "%"
1240 IF L=5 AND X1=SC+195 THEN POSITIO
N 19,5:? ")"
1250 POSITION M3,MM3:? MON$(MON,MON):P
OSITION KX,KY:? ">"
1260 POSITION 10,16:? "CHOOSE YOUR WEA
PON!";:GET #1,OWEP:OWEP=OWEP-48:POSITI
ON 10,16:? "                    "
1270 IF OWEP>6 OR OWEP<1 THEN 1260
1280 MIAD=1^1^1:HP=OHP(MON,OWEP):OWEP=
OWEP*2-1
1290 POSITION M3,MM3:? CON$(MON,MON):B
C=INT(2*RND(0)+1):CB=INT(2*RND(0)+1)
1300 POSITION M3,MM3:? CON$(MON,MON):B
C=INT(2*RND(0)+1):CB=INT(2*RND(0)+1)
1310 M3=M3+(BC=1)*(M3<22)-(BC=2)*(M3>1
6)
1320 MM3=MM3+(CB=1)*(MM3<11)-(CB=2)*(M
M3>5)
1330 POSITION WX,KY:? " ":WX=WX*JW
1340 POSITION M3,MM3:? MON$(MON,MON)
1350 J=STICK(0)
1360 IF STRIG(0)=0 OR JW=1 THEN 1480
1370 IF J=14 THEN POSITION KX,KY:? " "
:KY=KY-1
1380 IF J=13 THEN POSITION KX,KY:? " "
:KY=KY+1
1390 IF J=7 THEN POSITION KX,KY:? " ":
KX=KX+1
1400 IF J=11 THEN POSITION KX,KY:? " "
:KX=KX-1
1410 PE=PEEK(SC+KX+KY*40):IF PE=61 OR
PE=63 THEN SOUND 0,18,10,10:DMT=DMT+1^
1:SOUND 0,0,0,0:PT=PT+5:GOTO 1460
1420 IF PE=9 THEN SOUND 0,18,10,10:KEY
=1^1:SOUND 0,0,0,0:GOTO 1460
1430 IF PE=5 AND KEY THEN POSITION 4,1
6:? "YOU HAVE ESCAPED THE DUNGEON!!!":
FOR MIAD=0 TO 0 STEP 0:NEXT MIAD
1440 IF PE=5 THEN 1460
1450 IF PE>0 AND PE<>30 THEN SOUND 0,1
0,100,10:LI=LI-1:GOTO 1660
1460 POSITION KX,KY:? ">"
1470 GOTO 1290
1480 J=STICK(0)
1490 WEP=OWEP:PE=PEEK(SC+KX+KY*40):IF
PE>0 AND PE<>30 THEN SOUND 0,10,100,10
:LI=LI-1:GOTO 1660
1500 IF OWEP>6 THEN 1550
1510 JW=0
1520 IF J=7 AND KX<22 THEN WX=KX+1:GOT
O 1620
1530 IF J=11 AND KX>16 THEN WEP=WEP+1:
WX=KX-1:GOTO 1620
1540 GOTO 1290
1550 IF JW=1 THEN 1590
1560 IF J=7 AND KX<22 THEN RA=1:WX=KX:
GOTO 1610
1570 IF J=11 AND KX>16 THEN RA=-1:WX=K
X:GOTO 1610
1580 GOTO 1290
1590 PE1=PEEK(SC+WX+KY*40):IF PE1=61 O
R PE1=63 OR PE1=59 THEN WX=0:JW=0:GOTO
 1290
1600 IF PE1>0 THEN SOUND 0,16,14,10:JW
=0:THP=THP+HP*1^1:SOUND 0,0,0,0:IF THP
=30 THEN PT=PT+10:GOTO 1660
1610 WEP=WEP+(RA=-1):WX=WX+RA*(WX<23)*
(WX>15):JW=(WX<22)*(WX>16)
1620 PE1=PEEK(SC+WX+KY*40):IF PE1=61 O
R PE1=63 OR PE1=59 THEN WX=0:JW=0:GOTO
 1290
1630 IF PE1>0 THEN SOUND 0,16,14,10:JW
=0:THP=THP+HP*1^1:SOUND 0,0,0,0:IF THP
=30 THEN PT=PT+10:GOTO 1660
1640 POSITION WX,KY:? WEP$(WEP,WEP)
1650 GOTO 1290
1660 FOR W=5 TO 11:POSITION 16,W:? "
      ":NEXT W:POSITION 7,8:? LI;" ":SO
UND 0,0,0,0:CR=CR+1:POSITION 7,6:? PT
1670 POSITION 11,12:? DMT
1680 IF LI<1 THEN POSITION 15,15:? "GA
ME OVER":POSITION 7,16:? "PRESS BUTTON
 TO PLAY AGAIN":GOTO 2400
1690 J=STICK(0):UP=PEEK(X1-40):DO=PEEK
(X1+40):FO=PEEK(X1+1):BA=PEEK(X1-1)
1700 IF J=14 AND (UP=65 OR UP=81 OR UP
=68 OR UP=69 OR UP=87 OR UP=83 OR UP=1
24) THEN OW=1:ST=1:GOTO 1750
1710 IF J=13 AND (DO=65 OR DO=67 OR DO
=68 OR DO=88 OR DO=90 OR DO=83 OR DO=1
24) THEN OW=2:ST=-1:GOTO 1750
1720 IF J=11 AND (BA=65 OR BA=81 OR BA
=87 OR BA=82 OR BA=88 OR BA=90 OR BA=8
3) THEN OW=3:ST=1:GOTO 1810
1730 IF J=7 AND (FO=67 OR FO=68 OR FO=
69 OR FO=87 OR FO=82 OR FO=88 OR FO=83
) THEN OW=4:ST=-1:GOTO 1810
1740 GOTO 1690
1750 IF OW=1 THEN A=4:A1=11
1760 IF OW=2 THEN A=12:A1=5
1770 POSITION 16,A1+ST:? "       ":FOR
 W=A TO A1 STEP ST:POSITION 15,W:? "<<
<    <<<":POSITION 15,W+ST
1780 ? "<<<<<<<<<<":NEXT W
1790 POSITION 15,A1+ST:? "<<<    <<<"
1800 V=3*ST:A=7*(OW=1)+9*(OW=2):A1=9*(
OW=1)+7*(OW=2):POKE X1,11:GOTO 850
1810 IF OW=3 THEN A=15:A1=22
1820 IF OW=4 THEN A=23:A1=16
1830 POSITION A1+ST,4:? H$
1840 FOR W=A TO A1 STEP ST:POSITION W,
4:? A$
1850 POSITION W+ST,4:? G$:NEXT W:POSIT
ION A1+ST,7:? B$(19)
1860 V=3*ST:A=18*(OW=3)+20*(OW=4):A1=2
0*(OW=3)+18*(OW=4):POKE X1,11:GOTO 105
0
1870 RESTORE 2300+(L-1)*20
1880 FOR P=SC+188 TO SC+468 STEP 40:FO
R RT=P TO P+7:READ M:POKE RT,M:NEXT RT
:NEXT P:RETURN
1890 FOR W=A TO A1 STEP ST:POSITION 21
,W:?  "   ":NEXT W
1900 FOR W=A TO A1 STEP ST:POSITION 21
,W:? "<<<":POSITION 21,W+V:?  "   ":NEX
T W:GOTO 730
1910 FOR W=A TO A1 STEP ST:POSITION 15
,W:?  "   ":NEXT W
1920 FOR W=A TO A1 STEP ST:POSITION 15
,W:? "<<<":POSITION 15,W+V:?  "   ":NEX
T W:GOTO 730
1930 FOR W=A TO A1 STEP ST:POSITION 15
,W:?  "   ":NEXT W
1940 FOR W=A TO A1 STEP ST:POSITION 15
,W:? "<<<<<<":POSITION 15,W+V:?  "   ":
NEXT W:GOTO 730
1950 FOR W=A TO A1 STEP ST:POSITION 15
,W:?  "         ":NEXT W
1960 FOR W=A TO A1 STEP ST:POSITION 15
,W:? "<<<<<<<<<":POSITION 15,W+V:?  "
      ":NEXT W:GOTO 730
1970 FOR W=A TO A1 STEP ST:POSITION 21
,W:?  "   ":NEXT W
1980 FOR W=A TO A1 STEP ST:POSITION 18
,W:? "<<<<<<":POSITION 21,W+V:?  "   ":
NEXT W:GOTO 730
1990 FOR W=A TO A1 STEP ST:POSITION 15
,W:?  "         ":NEXT W
2000 FOR W=A TO A1 STEP ST:POSITION 15
,W:? "<<<    <<<":POSITION 15,W+V:?  "
      ":NEXT W:GOTO 730
2010 POSITION 15,A:? "<<<<<<<<<<"
2020 A1=11*(ST=1)+5*(ST=-1)
2030 FOR W=A TO A1 STEP ST:POSITION 15
,W:? "<    <":POSITION 15,W+ST:? "<
<<<<<<<":NEXT W
2040 POSITION 15,A:? "<<<<<<<<<<":IF N=
3 THEN QP=0:DM=0:GOTO 1210
2050 GOTO 1690
2060 IF (L>1 AND DMT<5) OR (L=1 AND DM
T<1) THEN 730
2070 CR=0:PT=PT+50:IF N=6 THEN L=L-1:D
MT=0:N=10:GOTO 2100
2080 IF N=10 THEN L=L+1:DMT=0:N=6
2090 FOR S=16 TO 32 STEP 4:SOUND 0,S,1
4,10:DL=EA*EA*EA:SOUND 0,0,0,0:EA=1^0:
NEXT S:GOTO 2110
2110 RL=7*(L=1)+9*(L=2)+15*(L=3)+15*(L
=4)+12*(L=5):POSITION 7,10:? L:POSITIO
N 11,12:? "0 "
2120 GOSUB 1870:GOTO 730
2130 FOR W=A TO A1 STEP ST:POSITION W,
4:? B$(19):NEXT W
2140 FOR W=A TO A1 STEP ST:POSITION W+
V,4:? B$(19):POSITION W,4:? G$(19):NEX
T W:GOTO 730
2150 FOR W=A TO A1 STEP ST:POSITION W,
4:? B$:NEXT W
2160 FOR W=A TO A1 STEP ST:POSITION W+
V,4:? B$:POSITION W,4:? A$:NEXT W:GOTO
 730
2170 FOR W=A TO A1 STEP ST:POSITION W,
4:? B$(19):NEXT W
2180 FOR W=A TO A1 STEP ST:POSITION W+
V,4:? B$(19):POSITION W,4:? F$:NEXT W:
GOTO 730
2190 FOR W=A TO A1 STEP ST:POSITION W,
4:? B$:NEXT W
```

```
2200 FOR W=A TO A1 STEP ST:POSITION W+
V,4:? B$:POSITION W,4:? G$:NEXT W:GOTO
 730
2210 FOR W=A TO A1 STEP ST:POSITION W,
7:? B$(10):NEXT W
2220 FOR W=A TO A1 STEP ST:POSITION W+
V,R:? B$(10):POSITION W,R:? F$:NEXT W:
GOTO 730
2230 FOR W=A TO A1 STEP ST:POSITION W,
10:? B$(19):NEXT W
2240 FOR W=A TO A1 STEP ST:POSITION W+
V,10:? B$(19):POSITION W,10:? G$(19):N
EXT W:GOTO 730
2250 POSITION A,4:? G$
2260 A1=22*(ST=1)+16*(ST=-1)
2270 FOR W=A TO A1 STEP ST:POSITION W,
4:? H$
2280 POSITION W+ST,4:? G$:NEXT W:POSIT
ION A,4:? G$:IF N=3 THEN QP=0:DM=0:GOT
O 1210
2290 GOTO 1690
2300 DATA 0,3,87,82,87,82,82,69,81,68,
65,69,90,3,69,124,124,10,124,124,81,82
,88,3,90,88,67,3,68,81,82,69,81,87
2310 DATA 82,69,90,67,81,67,90,83,82,8
8,82,87,88,69,3,124,81,3,87,3,69,124,9
0,88,88,82,88,82,88,67
2320 DATA 81,87,3,81,3,87,87,69,65,68,
124,124,65,68,3,124,124,6,124,65,3,90,
83,67,3,67,65,68,65,87,67,3,90,82
2330 DATA 67,90,3,124,81,68,81,87,82,8
7,88,67,124,10,124,3,82,3,69,81,68,124
,90,88,82,88,88,67,90,67,
2340 DATA 81,3,87,3,87,87,87,3,10,65,8
3,87,67,3,67,124,65,83,3,88,69,124,81,
3,3,67,124,81,3,68,65,69,0,81,88,68
2350 DATA 90,83,83,68,3,83,87,3,69,3,6
7,6,0,124,90,83,88,67,81,68,3,88,3,88,
3,82,67,3
2360 DATA 81,3,69,3,81,3,87,69,6,90,83
,83,88,67,65,3,90,3,67,65,82,3,90,69,8
1,69,81,3,69,90,87,3,3,68,65,67,124
2370 DATA 81,88,68,65,67,3,87,3,88,87,
3,10,81,67,124,65,69,65,68,90,3,82,3,6
7,90,3,67
2380 DATA 81,87,82,82,82,82,69,3,90,3,
81,3,69,3,90,67,81,67,124,81,83,88,82,
3,3,81,68,3,124,3,82,67,90,88,67,65
2390 DATA 67,124,81,3,81,69,0,90,69,65
,88,68,6,65,3,81,3,65,3,124,90,67,90,8
8,82,88,82,67
2400 IF STRIG(0)<>0 THEN 2400
2410 POKE 704,PEEK(710):POKE 53277,0:R
UN
```

# NIGHT RESCUE

```
100 GOTO 870
110 POKE 560,Z:POKE 561,D:SETCOLOR 0,1
2,6:SETCOLOR 1,13,10:SETCOLOR 2,3,2
120 SETCOLOR 3,8,4:SETCOLOR 4,Z,Z:LI=5
:RA=1:SP=11:POKE 756,CH/256:POKE 752,1

130 FOR AD=ND TO ND+33:READ P:POKE AD,
P:NEXT AD:POKE ND+26,PEEK(660):POKE ND
+27,PEEK(661):POKE 82,Z:POKE ND+7,D
140 POKE ND+12,D+1:POKE ND+18,D+6:POKE
 ND+21,D+7:POKE ND+24,D+8:FOR X=ND+64
TO ND+75:READ A:POKE X,A:NEXT X
150 POKE 656,Z:POKE 657,15:? "SCORE 0
   "
160 POKE 656,2:FOR JW=Z TO 19:POKE 657
,JW:? " ";:NEXT JW
170 POKE 656,2:POKE 657,7:? "!!!!!!"
180 FOR X=M TO ND+495:X=X+INT(10*RND(0
)+5):POKE X,48:POKE X+1,49:POKE X+2,54
:NEXT X
190 PE=0:RESTORE 690:FOR X=ND+1580 TO
ND+2307:READ A:POKE X,A:NEXT X
200 SP=SP+1:POKE 19,0:POKE 20,0
210 FOR W=1 TO 3:SOUND W,Z,Z,Z:NEXT W:
POKE 559,Z:POKE 77,Z:POKE 53278,Z:IF L
I=Z THEN 470
220 POKE 656,2:POKE 657,SP:? " ":SP=SP
-1
230 POKE ND+11,PE:POKE ND+17,PE:POKE N
D+20,PE:POKE ND+23,PE
240 POKE PLX,121:POKE PLY,85:POKE PLX+
1,32:POKE PLY+1,Z:POKE PLX+2,192:POKE
PLY+2,Z:POKE PLX+3,192:POKE PLY+3,66
250 PX=192:PY=66:BX=121:BY=85:BYA=0.25
:POKE 559,62:SOUND 1,180,2,1:SOUND 2,1
81,Z,1:SOUND 3,182,2,2
260 IF STICK(0)=11 AND  NOT STRIG(0) T
HEN BX=BX-(BX>109):POKE PLX,BX:GOTO 32
0
270 IF STICK(0)=7 AND  NOT STRIG(0) TH
EN BX=BX+(BX<133):POKE PLX,BX:GOTO 320

280 IF STICK(0)=11 AND PE>0 THEN POKE
756,CH1/256:PE=PE-1:PX=PX+4*(PX<=188):
GOTO 320
290 IF STICK(0)=7 AND PE<224 THEN POKE
 756,CH/256:PE=PE+1:PX=PX-4*(PX>=52):G
OTO 320
300 IF STICK(0)=14 THEN BYA=BYA-(BYA>-
5)
310 IF STICK(0)=13 THEN BYA=BYA+(BYA<5
)
320 POKE 53278,Z
330 POKE ND+11,PE:POKE ND+17,PE:POKE N
D+20,PE:POKE ND+23,PE
340 TI=INT((PEEK(19)*256+PEEK(20))/60)
:IF TI>300-SK*10 THEN POKE 19,0:POKE 2
0,0:GOTO 400
350 BY=BY+BYA:POKE PLY,BY:IF PY<60 OR
PY>90 THEN RA=-RA
360 PX=PX-(SK+3)*(PX<193):PY=2*RA+SK*R
A+PY:POKE PLX+3,PX:POKE PLY+3,PY:IF PX
<48 THEN PX=192
370 IF PEEK(53252)>0 OR PEEK(53260)>0
THEN 390
380 GOTO 260
390 IF PEEK(53252)=2 OR PEEK(53252)=8
THEN 420
400 LI=LI-1:FOR EA=1 TO 15 STEP 0.5:SO
UND Z,200+EA,4,16-EA:NEXT EA:SOUND Z,Z
,Z,Z:GOTO 210
410 GOTO 260
420 R=D+8-(BY<131):IF BY<115 THEN R=D+
6
430 MP=PE+INT((BX-45)/4+0.5)+256*R:IF
PEEK(MP)=131 THEN POKE MP,0:SC=SC+100:
GOTO 460
440 IF PEEK(MP)=11 AND PE=224 AND BX=1
33 AND BY>85 THEN FOR W=1 TO 3:SOUND W
,0,0,0:NEXT W:GOTO 1070
450 GOTO 400
460 FOR W=1 TO 10:SOUND Z,10,10,10:NEX
T W:SOUND Z,Z,Z,Z:POKE 656,Z:POKE 657,
21:? SC;:GOTO 260
470 POKE 656,2:POKE 657,SP:? " ":POKE
559,62:IF SC>HS THEN HS=SC
480 POKE 656,1:POKE 657,2:? "HIGH SCOR
E ";HS;:GOTO 1050
490 DATA 56,124,124,124,56,56,16,56,0,
0,0,0,48,252,48,204,52,28,52,28,255,23
5,255,235,0,0,0,0,255,235,255,235,255
500 DATA 235,255,235,255,235,255,235,0
,0,0,0,245,245,215,215,245,245,215,215
,245,245,215,215,85,40,85,40,255,255
510 DATA 255,40,245,245,215,215,0,0,0,
0,0,0,0,170,255,255,255,255,2,11,47,19
1,255,255,255,255,128,224,248,254,255
520 DATA 255,255,255,170,255,255,255,2
55,255,255,255,130,226,251,255,255,255
,255,255,130,235,255,255,255,255,255
530 DATA 255,130,139,239,255,255,255,2
55,255,0,0,0,0,0,0,40,190
540 DATA 255,255,255,255,215,105,105,2
15,192,192,192,192,192,192,192,64,255,
192,192,255,255,234,234,255
550 DATA 255,3,3,255,255,171,171,255,2
9,25,21,245,246,240,192,128,2,3,5,6,7,
23,31,30,0,0,64,64,64,80
560 DATA 208,208,208,144,80,124,124,60
,12,8,0,0,0,0,0,15,63,4,0,0,0,0,21,255
,255,68,0,0,0,0,170,240,252,64,255,0
570 DATA 21,41,41,21,1,10,255,128,80,8
4,85,85,84,170,192,0,0,3,87,64,0,0
580 DATA 0,0,0,0,2,11,47,191,0,0,0,0,1
28,224,248,254
590 DATA 3,0,0,192,213,1,0,0,255,2,5,2
1,85,85,21,170,255,0,84,104,104,84,64,
160
600 DATA 162,3,189,244,6,240,89,56,221
,240,6,240,83,141,254,6,106,141,255,6,
142,253,6,24,169,0,109,253,6,24,109
610 DATA 252,6,133,204,133,206,189,240
,6,133,203,173,254,6,133,205,189,248,6
,170,232,46,255,6,144,16,168,177
620 DATA 203,145,205,169,0,145,203,136
,202,208,244,76,87,6,160,0,177,203,145
,205,169,0,145,203,200,202,208,244
630 DATA 174,253,6,173,254,6,157,240,6
,189,236,6,240,48,133,203,24,138,141,2
53,6,109,235,6,133,204,24,173,253
640 DATA 6,109,252,6,133,206,189,240,6
,133,205,189,248,6,170,160,0,177,203,1
45,205,200,202,208,248,174,253,6,169
650 DATA 0,157,236,6,202,48,3,76,2,6,7
6,98,228,0,0,104,169,7,162,6,160,0,32,
92,228,96
660 DATA 28,62,127,127,127,127,62,28,2
8,8,28,28,9,25,127,127,255,255,24,24,8
,8
670 DATA 112,112,112,112,112,66,50,50,
112,112,68,0,51,112,112,112,112,69,0,56,69
,0,57,69,0,58
680 DATA 66,96,63,6,6,6,65,0,50,46,41,
39,40,52,0,50,37,51,35,53,37
690 DATA 0,0,0,0,6,5,131,0,0,0,131,5,6
,0,0,0,0,0,0,0,0,0,0,5,5,4,4,4,5,5,0
,0,0,0,0,0,0,0,0,0,0,9,9,9,9,9,9,9,9
700 DATA 0,0,0,0,0,0,9,131,9,0,0,0,0,0
,0,0,0,0,0,4,4,4,0,131,0,0,0,0,0,0,0,9
,131,0,0,0,0,7,8,8,8,10,10,8,8,8,8,10
710 DATA 10,10,10,10,8,8,8,8,8,8,8,8,1
0,10,10,10,8,10,10,10,8,8,8,8,8,10,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
720 DATA 0,0,0,0,0,6,6,4,0,0,0,0,131,0,4
,0,131,0,0,0,0,0,0,0,0,0,131,9,131,0,0
,0,0,0,0,0,0,0,0,0,0,131,0,9,9,0,0,0
730 DATA 0,0,0,131,9,9,9,131,0,0,0,55,
11,56,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,29,0,0,0,0,0,0,0,0,0,0,0,0
740 DATA 0,0,0,29,0,0,0,0,0,0,0,0,0,29
,0,0,0,0,0,0,0,0,0,0,36,38,0,6,6,6,5,131
,5,6,6,6,0,0,0,0,0,0,0,131
750 DATA 0,0,0,5,5,6,6,6,5,5,0,0,0,0,1
31,131,0,0,0,0,0,131,131,131,9,9,131,1
31,131
760 DATA 0,0,0,0,0,0,9,9,9,0,32,33,0,0,0
,36,38,0,131,5,6,6,6,6,6,131,0,36,38,0
,0,131,9,9,0,0
770 DATA 0,131,0,0,0,0,0,0,0,131,0,0,0
,0,131,0,0,0,10,10,10,8,8,0,10,0,0,0,1
31,0,0,0,0,0,10,10,0,0,131,0,0,7,131
780 DATA 0,0,0,0,0,0,0,0,0,0,0,36,38,0,0
,0,0,0,0,0,6,6,6,4,0,0,0,6,4,6,6,6,131
,0,0,32,33,0,0,0,0,9,9,9,0,36,38,0
790 DATA 36,38,0,0,0,0,0,0,0,9,9,9,9,0,0
,36,38,0,131,9,9,131,9,9,131,0,0,9,6,9
,0,0,32,33,0,0,0,0,0,36,38,0
800 DATA 55,11,12,15,26,30,13,56,131,5
5,11,11,12,28,27,13,11,56,131,55,12,30
,13,56,131,55,11,12,13,12,26
810 DATA 30,26,13,11,56,131,55,11,56,0
,34,42,131,6,6,6,6,6,6,6,6,6,131,43,44
,45,131,55,12,15,13,56,131,5,5,6,6,6
820 DATA 5,5,131,55,11,12,15,15,13,11,
56,131,0,9,9,9,9,9,9,9,131,43,44,45,
0,9,9,9,0,31,131
830 DATA 0,131,0,34,42,0,6,6,6,6,6,6,6
,6,131,34,42,131,0,9,9,9,0,131,7,8,8,7
,0,131,7,7,8,8,7,131,7,7,8,7,0,131
840 DATA 131,0,131,0,131,131,0,7,7,8
,8,8,7,0,131,7,7,7,7,8,8,8,8,8,7,131,4
3,44,45,131,43,44,45,0,34,42,131,0,43
850 DATA 44,45,131,0,6,6,6,6,6,0,131,0,6
,6,6,6,6,6,6,0,31,43,44,45,131,0,9,9,9
,0,34,42,131,34,42,0,43,44,45,0,131
860 DATA 9,9,9,9,131,0,34,42,131,9,9,9
,9,9,9,131,12,15,6,15,13,0,31,131,13
1,43,44,45,131,34,42,131,34,42,131
870 GRAPHICS 2:POKE 16,64:POKE 53774,6
4:DL=PEEK(560)+256*PEEK(561):D=PEEK(10
```

```
6)-14:ND=D*256:SETCOLOR 2,0,0
880 M=ND+256:M1=ND+511:Z=0:POKE DL+10,
6:POKE DL+11,6:POKE DL+12,6:POKE DL+13
,2
890 SETCOLOR 0,8,4:POSITION 4,4:? #6;"
NIGHT RESCUE":POSITION 5,8:? #6;"ANSCH
UETZ/WEISGERBER/ANSCHUETZ"
900 POSITION 4,6:? #6;"initializing":F
OR JW=M TO M1:POKE JW,0:NEXT JW
910 DIM CH$(33):CH$="!#$%&'()*+,-/:;<=
>?@ABDFJKLMPQVWX":CH=(PEEK(106)-4)*256
:FOR I=0 TO 543
920 POKE CH+I,PEEK(57344+I):NEXT I:FOR
 I=1 TO 33:CP=CH+(ASC(CH$(I))-32)*8:FO
R J=0 TO 7:READ A:POKE CP+J,A
930 NEXT J:NEXT I:CH1=(PEEK(106)-16)*2
56:PM=CH1:GOSUB 990:IF PEEK(CH1+10)=12
4 THEN 960
940 FOR I=0 TO 543:POKE CH1+I,PEEK(CH+
I):NEXT I
950 FOR X=384 TO 399:READ A:POKE CH1+X
,A:NEXT X:FOR X=432 TO 439:READ A:POKE
 CH1+X,A:NEXT X
960 POSITION 0,6:? #6;"press start to
begin"
970 IF PEEK(53279)<>6 THEN 970
980 RESTORE 670:POKE 559,0:GOTO 110
990 RESTORE 600:FOR I=1536 TO 1706:REA
D A:POKE I,A:NEXT I:FOR I=1774 TO 1787
:POKE I,0:NEXT I
1000 FOR I=PM+1023 TO PM+2047:POKE I,0
:NEXT I:FOR I=PM+1025 TO PM+1036:READ
A:POKE I,A:NEXT I
1010 FOR I=PM+1792 TO PM+1801:READ A:P
OKE I,A:NEXT I:FOR I=PM+1281 TO PM+179
1:POKE I,255:NEXT I
1020 POKE 704,134:POKE 705,0:POKE 706,
0:POKE 707,20:PLX=53248:PLY=1780:PLL=1
784:POKE 559,62:POKE 623,1
1030 POKE 1788,PM/256+4:POKE 53277,3:P
OKE 54279,PM/256:X=USR(1696):POKE PLL,
12:POKE PLL+1,255:POKE PLL+2,255
1040 POKE PLL+3,10:POKE 53257,3:POKE 5
3258,3:POKE 53259,1:RESTORE 590:RETURN

1050 IF PEEK(53279)<>6 THEN 1050
1060 LI=5:SK=0:RA=1:SP=11:SC=0:POKE 55
9,0:FOR JW=M TO M1:POKE JW,0:NEXT JW:P
OKE 756,CH/256:GOTO 150
1070 SOUND 1,250,10,10:SOUND 2,254,10,
10:FOR W=1 TO 20:NEXT W:SOUND 1,0,0,0:
SOUND 2,0,0,0
1080 SK=SK+1:SC=SC+SK*(1000-TI):POKE 6
56,0:POKE 657,21:? SC;:FOR EA=M TO ND+
455:POKE EA,0:NEXT EA:POKE 559,0
1090 FOR JW=ND+456 TO M1:POKE JW,0:NEX
T JW:GOTO 180
```

# OVERFLOW

```
10 REM OVERFLOW
20 REM BY ANSCHUETZ/WEISGERBER/ANSCHUE
TZ
30 REM (c) 1985, ANTIC PUBLISHING
40 TUNL=50:WATR=148:MT=0
50 POKE 106,PEEK(740)
60 MEM=PEEK(106):DL=(MEM-5)*256:CH=(ME
M-4)*256:JW=MEM-12:EA=MEM-11:RA=(MEM-9
)*256:S=(MEM-7)*256
70 GRAPHICS 2:POKE 709,50:DIM Q(10),A$
(46),B$(16):POKE 755,0:POSITION 6,4:?
#6;"OVERFLOW"
80 POKE 710,30:POKE 712,30:? :? "    A
nschuetz/Weisgerber/Anschuetz":GOTO 41
0
90 POKE 709,TUNL:POKE 710,WATR:POKE 71
2,0:POKE 707,TUNL
100 POKE 89,JW:POKE 756,MEM-4:POKE 559
,62:POKE 560,0:POKE 561,MEM-5:POKE 532
51,120:POKE 53248,123
110 ? :? :? :? :FOR X=1 TO 10:Q(X)=INT
(RND(0)*2)*2-1:NEXT X:SC=0:POKE 656,0:
POKE 657,16:? "OVERFLOW"
120 POKE 657,4:? "SCORE:0    ";:POKE 6
57,26:? "HIGH:";HC;:YX=33:POKE 88,YX:P
OSITION 0,0:? #6;A$(17,30)
130 QL=DL+1:C=0:FOR X=DL+4 TO DL+31 ST
EP 3:Y=INT(RND(0)*11)+101:Y=Y+(Y/2<>IN
T(Y/2)):POKE X,Y:NEXT X
140 C=C+1:QL=QL+3:POKE QL,PEEK(QL)+Q(C
)
150 POKE 53278,0:IF QL>DL+28 THEN QL=D
L+1:C=0
160 ST=STICK(0):IF ST=14 THEN YX=YX-16
*(YX>33):POKE 88,YX:POSITION 0,0:? #6;
A$(17,46)
170 POKE 709,TUNL:POKE 710,WATR
180 IF ST=13 THEN YX=YX+16*(YX<177):PO
KE 88,YX-16:POSITION 0,0:? #6;A$(1,30)

190 IF ST=7 AND J=0 THEN PLX=133:PLY=Y
X+3:J=4:POKE 89,EA:POKE 88,YX+3:POSITI
ON 0,0:? #6;B$(1,8):POKE 89,JW
200 IF ST=11 AND J=0 THEN PLX=115:PLY=
YX+3:J=-4:POKE 89,EA:POKE 88,YX+3:POSI
TION 0,0:? #6;B$(9,16):POKE 89,JW
210 IF PEEK(53255)>3 THEN POKE 707,WAT
R:GOTO 360
220 IF J<>0 THEN 240
230 GOTO 140
240 IF PEEK(53253)>3 THEN 270
250 POKE 53249,PLX:PLX=PLX+J
260 IF PEEK(53253)<4 THEN 290
270 HQ=(PLY-20)/16:IF SGN(Q(HQ))<>SGN(
J) THEN 330
280 SOUND 0,100,10,10:SC=SC+10:Q(HQ)=-
Q(HQ):POKE 657,10:? SC;:GOTO 310
290 IF PLX>200 OR PLX<48 THEN 330
300 GOTO 140
310 POKE 77,0:IF SC=250 THEN SOUND 0,6
0,10,12:GOTO 350
320 IF SC=500 THEN SOUND 0,60,10,12:PO
KE 53259,3:POKE 53251,112:FOR X=1 TO 1
00:NEXT X
330 POKE 53249,0:J=0:POKE 89,EA:POKE 8
8,PLY:POSITION 0,0:? #6;A$(1,8):POKE 8
9,JW:POKE 53278,0
340 SOUND 0,0,0,0:GOTO 140
350 FOR X=1 TO 10:Q(X)=Q(X)*2:NEXT X:F
OR X=DL+4 TO DL+31 STEP 3:Y=PEEK(X):PO
KE X,Y+(Y/2=INT(Y/2)):NEXT X:GOTO 330
360 J=0:POKE 656,3:POKE 657,9:? "PRESS
 <START> TO BEGIN";:IF SC>HC THEN HC=S
C
370 YC=YC+(YC<255)-255*(YC=255):POKE 7
04,YC:IF PEEK(53279)<>6 THEN 370
380 FOR X=DL+4 TO DL+31 STEP 3:POKE X,
108:NEXT X:POKE 89,EA:POKE 88,PLY:POSI
TION 0,0:? #6;A$(1,8):POKE 53251,120
390 POKE 704,198
400 POKE 89,JW:POKE 88,YX:POSITION 0,0
:? #6;"                ":POKE 707,TUNL:
POKE 53249,0:POKE 53259,1:GOTO 110
410 FOR X=0 TO 511:POKE X+CH,PEEK(X+57
344):NEXT X:FOR X=8 TO 23:READ A:POKE
X+CH,A:NEXT X
420 FOR X=DL TO DL+2:POKE X,112:NEXT X
:FOR X=DL+3 TO DL+30 STEP 3:POKE X,69:
POKE X+1,108:POKE X+2,S/256:NEXT X
430 POKE DL+33,66:POKE DL+34,PEEK(660)
:POKE DL+35,PEEK(661):FOR X=DL+36 TO D
L+38:POKE X,2:NEXT X:POKE DL+39,65
440 FOR X=RA+32 TO RA+191:POKE X,255:N
EXT X:FOR X=JW*256 TO RA:POKE X,0:NEXT
 X:FOR X=1 TO 46:A$(X,X)=" ":NEXT X
450 FOR X=17 TO 30:READ A:A$(X)=CHR$(A
+32):NEXT X:A$(46,46)=" ":FOR X=1 TO 1
6:READ A:B$(X)=CHR$(A):NEXT X
460 POKE 623,1:POKE 54279,MEM-16:POKE
53277,3:POKE 704,198:POKE 705,198:POKE
 707,TUNL:POKE 53259,1
470 FOR X=5 TO 5+108:POKE X,2:POKE X+1
47,2:NEXT X:FOR X=5+109 TO 5+146:POKE
X,1:NEXT X:SOUND 1,160,8,2:GOTO 90
480 DATA 84,170,170,170,170,170,170,69
,84,255,255,255,255,255,255,69
490 DATA 14,14,14,4,31,31,14,14,14,14,
10,10,10,10,33,35,35,255,255,35,35,33,
160,128,128,255,255,128,128,160
```

# PHOENIX

```
10 GRAPHICS 17:COLOR 0:SETCOLOR 0,0,0:
POKE 756,226:POKE 752,1
18 PG=0:SC=0
19 R=5:RR=4:X=9:Y=18:IF PG>9 THEN 500
20 SOUND 1,0,0,0:GOSUB 30
25 GOTO 60
30 POSITION R,RR:? #6;"◘":POSITION R-1
,RR-1:? #6;"◥":POSITION R+1,RR-1:? #6;
"◢"
40 POSITION R-1,RR+1:? #6;"▙":POSITION
 R+1,RR+1:? #6;"▟"
50 POSITION X,Y:? #6;"|":POSITION X,Y-
1:? #6;"●":POSITION X-1,Y:? #6;"◢":POS
ITION X+1,Y:? #6;"◣"
55 RETURN
60 POSITION R,RR:? #6;" ":POSITION R-1
,RR-1:? #6;" ":POSITION R+1,RR-1:? #6;
" "
61 POSITION R-1,RR+1:? #6;" ":POSITION
 R+1,RR+1:? #6;" "
62 SD=INT(RND(0)*4)+1:IF SD=1 THEN R=R
+2:IF R>18 THEN R=R-4
63 IF SD=2 THEN R=R-2:IF R<4 THEN R=R+
4
64 IF SD=3 THEN RR=RR+2:IF RR>10 THEN
RR=RR-4
65 IF SD=4 THEN RR=RR-2:IF RR<2 THEN R
R=RR+4
66 POSITION R,RR:? #6;"◘":POSITION R-1
,RR:? #6;"▀":POSITION R+1,RR:? #6;"▀"
67 POSITION R-1,RR+1:? #6;"▙":POSITION
 R+1,RR+1:? #6;"▟"
68 POSITION R,RR:? #6;" ":POSITION R-1
,RR:? #6;" ":POSITION R+1,RR:? #6;" "
69 POSITION R-1,RR+1:? #6;" ":POSITION
 R+1,RR+1:? #6;" "
70 POSITION X,Y:? #6;" ":POSITION X,Y-
1:? #6;" ":POSITION X-1,Y:? #6;" ":POS
ITION X+1,Y:? #6;" "
71 IF STICK(0)=7 THEN X=X+2:IF X>18 TH
EN X=3
72 IF STICK(0)=11 THEN X=X-2:IF X<2 TH
EN X=17
80 IF STRIG(0)=0 THEN 100
90 GOTO 20
100 GOSUB 30:HU=16:SC=SC-1:SOUND 1,INT
(RND(0)*100)+100,8,14
101 POSITION X,HU:? #6;"↑":POSITION X,
HU:? #6;" ":HU=HU-1:IF X=R AND HU=RR T
HEN 150
102 IF HU<1 THEN 20
105 GOTO 101
120 GOTO 20
150 SOUND 1,0,0,0:PG=PG+1:POSITION R,R
R:? #6;" ":POSITION R-1,RR-1:? #6;" ":
POSITION R+1,RR-1:? #6;" ":SC=SC+11
151 POSITION R-1,RR+1:? #6;" ":POSITIO
N R+1,RR+1:? #6;" "
160 POSITION R,RR:? #6;"◘":POSITION R-
1,RR+1:? #6;"◢":POSITION R+1,RR+1:? #6
;"◣"
170 POSITION R-1,RR-1:? #6;"▉":POSITIO
N R+1,RR-1:? #6;"▉"
175 POSITION R,RR:? #6;" ":POSITION R-
1,RR+1:? #6;" ":POSITION R+1,RR+1:? #6
;" "
180 POSITION R-1,RR-1:? #6;" ":POSITIO
N R+1,RR-1:? #6;" "
181 FOR A=256 TO 253 STEP -1:SOUND 2,A
,10,10:NEXT A
182 RR=RR+1:IF RR<16 THEN 190
183 SOUND 2,0,0,0
184 POSITION R,RR:? #6;" ":POSITION R-
1,RR+1:? #6;" ":POSITION R+1,RR+1:? #6
;" "
185 POSITION X,Y:? #6;" ":POSITION X,Y
-1:? #6;" ":POSITION X-1,Y:? #6;" ":PO
SITION X+1,Y:? #6;" "
186 POSITION R-1,RR-1:? #6;" ":POSITIO
N R+1,RR-1:? #6;" ":GOTO 19
190 GOTO 160
500 GRAPHICS 2+16
501 ? #6;"     SCORE = ";SC
502 POSITION 0,9
505 IF SC>94 THEN ? #6;" PHOENIX COMMA
NDER":GOTO 1000
510 IF SC>84 THEN ? #6;"    BIRD CAPTA
IN":GOTO 1000
515 IF SC>79 THEN ? #6;"  BUTTERFLY NE
TTER":GOTO 1000
520 ? #6;"    BIRD WATCHER"
1000 FOR W=1 TO 500:NEXT W:RUN
```

# PIRACY

```
50 GRAPHICS 0:SETCOLOR 2,0,0:POKE 752,
1:POKE 559,0
55 POSITION 3,4:? "▛▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▜":POSI
TION 3,5:? "▌▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚▐
▐▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚▐
▙▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▟▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚▗▚
```

```
W:GRAPHICS 18:? #6;"YOU HAVE CAPTURED
  ALL 16 TREASURES.   YOU HAVE COLLECT
ED"
12005 ? #6;D;" DOLLARS.":FOR W=1 TO 10
00:NEXT W:RUN
13000 FOR W=0 TO 3:SOUND W,0,0,0:NEXT
W
13002 GRAPHICS 18:? #6;"THE ENEMY HAS
      DESTROYED YOU BY    RAMMING YOUR
 SHIP.   YOU HAVE COLLECTED"
13005 ? #6;D;" DOLLARS.":FOR W=1 TO 10
00:NEXT W:RUN
14000 FOR W=0 TO 3:SOUND W,0,0,0:NEXT
W
14002 GRAPHICS 18:? #6;"YOU RAN OUT OF
 FUEL AND COULD NOT       CONTINUE. YO
U       COLLECTED"
14005 ? #6;D;" DOLLARS.":FOR W=1 TO 10
00:NEXT W:RUN
```

# PIRACY II

```
100 E=0:R=1:T=9:K=256:H=559:P=62:G=3:U
=24:Y=560:M=Y+R:V=14:J=PEEK(106):GOSUB
 1240:GRAPHICS E:POKE H,E:GOSUB 1230
110 DIM BA(T),COUNT(T),SHIP(T),F(T),IS
(T),XP(T),GONE(10):ISNUM=HS:XXE=0:F=R:
ISF=R:DE=45:FOR X=A3 TO T
120 GONE(X)=R:NEXT X:POKE B1,E:POKE B2
,E:POKE 1776,E:GOSUB 130:GOTO 140
130 RESTORE 1290:FOR X=704 TO 712:READ
 A:POKE X,A:NEXT X:RETURN
140 MIAD=0.9:BR=20:SET=R:LI=T+R:FOR X=
E TO T:SHIP(X)=E:NEXT X:DL=(J-V)*K:IF
RU THEN 160
150 FOR X=DL TO DL+JW+R:READ A:POKE X,
A:NEXT X:POKE DL+JW+R,J-V
160 RESTORE 1320:FOR X=E TO T:READ A:I
S(X)=A:COUNT(X)=E:NEXT X:FOR X=E TO HS
:GONE(X)=E:NEXT X
170 POKE Y,E:POKE M,J-V:S=J-40:CH=(J-A
1)*K:IF RU THEN GOSUB 200:GOTO 240
180 Q=(J-16)*K+150:FOR X=Q TO Q+33:REA
D A:POKE X,A:NEXT X
190 FOR X=1536 TO 1733:READ A:POKE X,A
:NEXT X:GOSUB 200:GOTO 220
200 FOR X=1736 TO 1747:POKE X,128:NEXT
 X
210 Z=S:FOR X=1756 TO 1767:POKE X,Z:Z=
Z+R:NEXT X:RETURN
220 FOR I=E TO 767:POKE CH+I,PEEK(5734
4+I):NEXT I:FOR I=A1 TO 239:READ A:POK
E CH+I,A:NEXT I
230 FOR I=288 TO 327:READ A:POKE CH+I,
A:NEXT I
240 POKE 756,CH/K:POKE 207,E:POKE 208,
S-24:POKE 1769,S+22:IF RU THEN 270
250 RESTORE 1650:Q=(J-A3)*K:FOR X=Q TO
 Q+44:READ A:POKE X,A:NEXT X
260 FOR X=1783 TO 1790:READ A:POKE X,A
:NEXT X:FOR I=240 TO 287:POKE CH+I,E:N
EXT I
270 PM=J-16:PMB=PM*K:POKE 623,33:POKE
CL-R,G:POKE 54279,PM:POKE CL,E
280 IF RU THEN X=USR(PMB+150):GOTO 320

290 RESTORE 1680:FOR X=PMB TO PMB+141:
READ A:POKE X,A:NEXT X:POKE PMB+A2,J-1
6:X=USR(PMB+150)
300 FOR X=PMB+1224 TO PMB+1241:READ A:
POKE X,A:NEXT X:FOR X=PMB+1479 TO PMB+
1503:READ A:POKE X,A:NEXT X
310 FOR X=PMB+1660 TO PMB+1668:READ A:
POKE X,A:NEXT X
320 FOR X=53248 TO 53252:POKE X,E:NEXT
 X:POKE 53250,125:POKE 208,J-V
330 FOR SS=E TO G:RESTORE 1530:XP(SS)=
K*(INT(RND(E)*27)+S-16):FOR A=E TO G:F
OR B=E TO 7
340 READ X:POKE XP(SS)+IS(SS)+B+K*A,X:
NEXT B:NEXT A:SHIP(SS)=XP(SS)+IS(SS)+A
1:POKE SHIP(SS),167:NEXT SS
350 IF RU THEN 430
360 FOR SS=(S-U)*K TO (S+22)*K STEP K:
POKE SS+22,A4:POKE SS+23,A4:POKE SS+23
2,A4:POKE SS+233,A4:NEXT SS
370 FOR SS=(S-U)*K+1536 TO (S-U)*K+179
1:POKE SS,A4:POKE SS+8448,A4:NEXT SS
380 FOR SS=(S-U)*K TO (S+22)*K STEP K:
FOR WW=E TO 21:POKE SS+WW,A5:POKE SS+2
34+WW,A5:NEXT WW:NEXT SS
390 FOR SS=(S-U)*K TO (S-U)*K+1535:POK
E SS,A5:POKE SS+10240,A5:NEXT SS
400 RESTORE 1760:Q=(J-A3)*K:FOR X=Q+50
 TO Q+127:READ A:POKE X,A:NEXT X:POKE
Q+56,Q/K:POKE Q+76,Q/K
410 POKE Q+86,Q/K:POKE Q+100,Q/K:FOR X
=Q+190 TO Q+201:READ A:POKE X,A:NEXT X
:FOR X=Q+210 TO Q+225:READ A
420 POKE X,A:NEXT X:FOR X=Q+170 TO Q+1
85:READ A:POKE X,A:NEXT X
430 POKE CL,E:POKE 54,E
440 RESTORE 1530:XP(ISNUM)=K*(INT(RND(
E)*27)+S-16):FOR A=E TO G:FOR B=E TO 7

450 READ X:POKE XP(ISNUM)+IS(ISNUM)+B+
K*A,X:NEXT B:NEXT A:SHIP(ISNUM)=XP(ISN
UM)+IS(ISNUM)+A1:POKE SHIP(ISNUM),167
460 X=USR(1536):POKE 1770,E:POKE 1771,
E:POKE CL,E:POKE 54,E:POKE H,P:SOUND E
,50,A1,L
470 FOR SE=E TO ISNUM:WQ=INT(RND(E)*G)
-R:WR=K*(WQ=R)-K*(WQ=-R)+E*(WQ=E):SHNE
W=SHIP(SE):DS=INT(RND(E)*L)
480 SHNEW=SHNEW+WQ*(DS=R)+WR*(DS=E)
490 IF PEEK(SHNEW)=40 AND COUNT(SE)=E
THEN POKE SHIP(SE),40:POKE SHNEW,167:S
HIP(SE)=SHNEW
500 NEXT SE:HIT=PEEK(53254):POKE 77,E:
IF HIT=8 OR HIT=10 AND  NOT SF THEN 58
0
510 IF HIT=8 OR HIT=R OR HIT=10 OR HIT
=3 OR HIT=11 OR HIT=9 THEN 530
520 IF HIT=4 OR HIT=6 AND  NOT ISF THE
N N=PEEK(1740)+20:GOSUB 760:GOTO 800
530 IF HIT<>2 AND HIT<>E THEN LI=LI-R:
GOSUB 1220:GOSUB 1190:GOSUB 570:POKE H
,P
540 IF STRIG(E)=E THEN POKE H,E:POKE S
3250,E:POKE 54,R:GOSUB 1090:POKE H,P
550 IF XXE=E THEN POKE CH+321,E:POKE C
H+322,40:POKE CH+325,E:POKE CH+326,40:
XXE=R:GOTO 470
560 POKE CH+321,40:POKE CH+322,130:POK
E CH+325,40:POKE CH+326,130:XXE=E:GOTO
 470
570 POKE H,E:POKE 53250,E:GOSUB 200:PO
KE CL,E:POKE 53250,125:POKE 54,E:RETUR
N
580 SOUND E,E,E,E:GRAPHICS E
590 POKE H,E:GOSUB 1230:POKE 756,CH/K:
POKE 1782,100:POKE 1777,E:POKE 1778,G:
POKE 1779,G:POKE 1780,E
600 POKE 207,E:POKE 208,PM+G:POKE 5325
6,G:POKE 53257,G:FOR X=PMB+768 TO PMB+
1023:POKE X,E:NEXT X:POKE 53250,E
610 DL=PEEK(Y)+K*PEEK(M)+HS:POKE DL-R,
69:FOR I=L TO 12:POKE DL+I,A3:NEXT I:R
ESTORE 1540
620 POKE CL,E:POKE DL+13,65:SC=PEEK(88
)+K*PEEK(89):FOR A=R TO G:FOR B=E TO 3
9:READ Z:POKE B+SC+JW*A,Z:NEXT B
630 NEXT A:FOR W=E TO A1:BA(W)=SC+163+
(W*4):F(W)=E:NEXT W:XZ=INT(RND(E)*JW):
POKE SC+XZ,152:OZ=XZ:POKE H,P
640 POKE CL,E:POKE 1771,E:X=USR(PMB):P
OKE 54,E
650 CZ=INT(RND(E)*T):F(CZ)=L-(F(CZ)=R)
:IF F(CZ)=L THEN SOUND E,200,L,10:POKE
 BA(CZ),37:F(CZ)=R
660 IF RND(E)>MIAD THEN XZ=INT(RND(E)*
40):POKE SC+OZ,E:POKE SC+XZ,152:OZ=XZ
670 FOR W=R TO BR:NEXT W:POKE 77,E:SOU
ND E,E,E,E
680 FOR CZ=E TO A1:IF F(CZ) THEN BA(CZ
)=BA(CZ)+JW:IF BA(CZ)>SC+480 THEN POKE
 BA(CZ)-JW,E:F(CZ)=E:BA(CZ)=BA(CZ)-320

690 IF F(CZ) THEN POKE BA(CZ)-JW,E:POK
E BA(CZ),37
700 IF PEEK(1776) THEN POKE H,E:ISF=E:
SF=R:COUNT(NUM)=R:POKE SHIP(NUM),40:NU
M=NUM+(NUM<T):GOTO 730
710 IF PEEK(53252) OR PEEK(53253) THEN
 GOSUB 1220:LI=LI-R:GOSUB 1190:GOTO 58
0
720 NEXT CZ:GOTO 650
730 SCORE=SCORE+100*SET
740 POKE H,E:POKE 1771,R:POKE 53248,E:
POKE 53249,E:POKE 53252,E:POKE 54,R:PO
KE 208,J-V
750 X=USR(1536):POKE Y,E:POKE M,J-V:PO
KE 1776,E:POKE CL,E:GOSUB 130:GOSUB 57
0:GOTO 460
760 GH=(N>80 AND N<102)+L*(N>E AND N<A
4)+G*(N>211 AND N<255)+HS*(N>A4 AND N<
58)
770 GH=GH+A3*(N>190 AND N<212)+A2*(N>1
02 AND N<124)+7*(N>58 AND N<80)+A1*(N>
124 AND N<146)+T*(N>168 AND N<190)
780 IF GONE(GH)=R THEN POP :LI=LI-R:GO
SUB 1220:GOSUB 1190:GOSUB 570:POKE H,P
:SOUND E,50,A1,L:GOTO 470
790 RETURN
800 SOUND E,E,E,E:POKE 53277,E
810 GRAPHICS E:POKE 559,E:GOSUB 1230:P
OKE 756,CH/K:POKE 207,240:POKE 208,CH/
K:POKE 1791,DE:RESTORE 1570
820 DL=PEEK(Y)+K*PEEK(M)+HS:POKE DL-R,
69:FOR I=L TO 12:POKE DL+I,A3:NEXT I:S
C=PEEK(88)+K*PEEK(89)
830 POKE 53250,E:POKE CL,E:POKE DL+13,
65:FOR I=R TO 31:READ A,B,C,D,EE,F:POK
E SC+A,30
840 POKE SC+B,31:POKE SC+C,32:POKE SC+
D,33:POKE SC+EE,34:POKE SC+F,35:NEXT I

850 FOR W=SC TO SC+RA:POKE W,EA:POKE W
+440,EA:NEXT W:FOR W=SC+JW TO SC+400 S
TEP JW:POKE W,EA:POKE W+R,EA
860 POKE W+EA,EA:POKE W+RA,EA:NEXT W
870 T1=PEEK(B1):T2=PEEK(B2):POKE B1,E:
POKE B2,(PMB+1792)/K:POKE CL-R,G:POKE
1773,56:POKE 1772,194
880 POKE SC+230,156:POKE SC+231,157:PO
KE 53251,56:POKE 53252,E:POKE 1770,R:X
=USR((J-A3)*K+50)
890 FOR X=PMB+1792 TO PMB+2047:POKE X,
E:NEXT X:POKE 53248,E:POKE 53249,E:POK
E 54,E:POKE H,P
900 POKE 53768,U:FOR W=E TO A2 STEP L:
POKE 53761+W,168:NEXT W:POKE 53760,240
```

```
:POKE 53764,252:POKE 53762,28
910 POKE 53766,49:POKE 1770,E
920 POKE 77,E:FOR Q=30 TO 35:X=USR((J-
A3)*K,(Q-30)*A1):GOSUB 930:NEXT Q:GOTO
 960
930 IF PEEK(53255)=A1 THEN GOSUB 1210:
POKE H,E:POP :ISF=R:SF=E:SCORE=SCORE+2
50*SET:GOTO 990
940 IF PEEK(53255) THEN GOSUB 1210:POK
E H,E:POKE 1770,R:POP :GOTO 970
950 RETURN
960 FOR Q=34 TO 31 STEP -R:X=USR((J-A3
)*K,(Q-30)*A1):GOSUB 930:NEXT Q:GOTO 9
20
970 POKE H,E:LI=LI-R:GOSUB 1190:FOR X=
PMB+1792 TO PMB+2047:POKE X,E:NEXT X:P
OKE 1773,56
980 POKE 1772,194:POKE CL,E:POKE 1770,
E:POKE H,P:GOTO 900
990 POKE XP(GH)+IS(GH)+262,G:GONE(GH)=
R:FOR WE=E TO T:IF GONE(WE)=E THEN POP
 :GOTO 1080
1000 NEXT WE:FOR WE=E TO ISNUM:GONE(WE
)=E:COUNT(WE)=E:POKE XP(WE)+IS(WE)+262
,10:POKE XP(WE)+IS(WE)+A1,167:NEXT WE
1010 FOR WE=E TO ISNUM:SHIP(WE)=XP(WE)
+IS(WE)+A1:NEXT WE
1020 ISNUM=ISNUM+R:GONE(ISNUM)=E:IF IS
NUM=10 THEN ISNUM=T:FD=R
1030 SCORE=SCORE+1000*SET:DE=DE-A3*(DE
>25):MIAD=MIAD-0.05*(MIAD>0.7):BR=BR-A
3*(BR>E):SET=SET+R:NUM=E
1040 POKE 1770,R:POKE B1,T1:POKE B2,T2
:POKE 207,E:POKE 53251,E
1050 POKE 1771,R:POKE 53248,E:POKE 532
49,E:POKE 53252,E:POKE 54,R:POKE 208,J
-V:POKE Y,E:POKE M,J-14
1060 POKE 1776,E:POKE CL,E:GOSUB 130:G
OSUB 570:POKE 53250,E:GOSUB 1090:IF FD
 THEN FD=E:GOTO 460
1070 GOTO 440
1080 POKE 1770,R:POKE B1,T1:POKE B2,T2
:POKE 207,E:POKE 53251,E:GOTO 740
1090 SOUND E,E,E,E:GRAPHICS 18:POKE H,
E:GOSUB 1230:POSITION H5,R:? #6;"SCORE
: ";SCORE:POSITION A2,H5
1100 ? #6;"ships: ";LI:POSITION A3,7:?
 #6;"MISSION ";SET:POSITION H5,10:? #
6;"press trigger"
1110 IF  NOT LI THEN POSITION H5,10:?
#6;"  GAME OVER  "
1120 POKE H,P:FOR W=R TO 100:NEXT W:IF
  NOT LI THEN 1160
1130 IF STRIG(E) THEN POKE 77,E:GOTO 1
130
1140 POKE H,E:POKE Y,E:POKE M,J-V:POKE
 54,E:POKE 53250,125
1150 GOSUB 130:POKE 756,CH/K:SOUND E,5
0,A1,L:RETURN
1160 IF STRIG(E)=E THEN FL=R
1170 IF FL THEN FL=E:POKE 1770,R:POKE
1771,R:POKE 54,R:FOR X=53248 TO 53252:
POKE X,E:NEXT X:POKE CL,E:RUN
1180 POKE 77,E:GOTO 1160
1190 IF  NOT LI THEN POKE H,E:POKE CL-
R,E:POP :GOTO 1090
1200 RETURN
1210 FOR W=E TO G:SOUND W,E,E,E:NEXT W
:RETURN
1220 SOUND R,100,L,10:FOR W=R TO 20:NE
XT W:SOUND R,E,E,E:RETURN
1230 POKE 16,64:POKE 53774,64:RETURN
1240 GRAPHICS R:L=2:EA=38:RA=39:SETCOL
OR L,E,E:GOSUB 1230:54=1774:DL=PEEK(Y)
+K*PEEK(M):POKE DL+V,23:POKE DL+18,L
1250 JW=40:POKE 54276,4:SETCOLOR E,8,4
:POSITION 7,T:? #6;"PIRACY II":POSITIO
N 4,11:? #6;"press start to begin"
1260 RU=PEEK(1536):A1=8:A2=6:A3=5:H5=4
:A4=36:A5=155:CL=53278:POSITION T,13:?
 #6;"ANSCHUETZ/WEISGERBER/ANSCHUETZ"
1270 B1=204:B2=B1+R:POKE 77,E:IF PEEK(
53279)<>6 THEN 1270
1280 RETURN
1290 DATA 52,136,50,198,214,134,28,0,1
36
1300 DATA 112,112,112,117,128,96,117,1
28,97,117,128,98,117,128,99,117,128,10
0,117,128,101,117,128,102
1310 DATA 117,128,103,117,128,104,117,
128,105,117,128,106,85,128,107,65,0,50

1320 DATA 157,91,25,223,47,201,113,69,
135,179,104,160,0,177,207,201,40,240,1
2
1330 DATA 201,36,240,8,201,155,240,4,1
69,40,145,207,200,208,235,230,208,165,
208,205,233,6,208,224,96
1340 DATA 104,169,7,160,11,162,6,32,92
,228,96,173,238,6,240,3,76,98,228,216,
173,120,2,41,1,208,23,198,205
1350 DATA 165,205,16,17,169,15,133,205
,162,0,222,220,6,232,138,201,12,144,24
7,176,92,173,120,2,41,2,208,25,230
1360 DATA 205,165,205,201,16,208,17,16
9,0,133,205,162,0,254,220,6,232,138,20
1,12,144,247,176,60
1370 DATA 173,120,2,41,4,208,25,230,20
4,165,204,201,4,208,17,169,0,133,204,1
62,0
1380 DATA 222,200,6,232,138,201,12,144
,247,176,28,173,120,2,41,8,208,21,198,
204,165,204,16,15,169,3,133,204,162,0
1390 DATA 254,200,6,232,138,201,12,144
,247,162,0,138,10,134,206,24,101,206,1
05,4,168,189,200,6,145,207,200
1400 DATA 189,220,6,145,207,232,138,20
1,12,144,229,165,205,141,5,212,165,204
,141,4,212,173,6,208,240,9,201,2
1410 DATA 240,5,169,1,141,238,6,76,98,
228
1420 DATA 0,0,0,0,0,0,60,255,15,63,15,
3,3,15,63,255,255,255,255,255,255,255,
255,255,0,255,255,255,255,255,255,255
1430 DATA 0,240,252,240,240,240,255,25
5,0,3,15,15,3,0,0,0,255,255,255,255,25
5,255,63,3,255,255,255,255,255,255,255

1440 DATA 195,215,85,85,150,235,239,23
9,239,255,255,255,255,255,87,238,234,2
55,255,255,255,255,255,255,255,63,0
1450 DATA 15,63,255,255,63,15,0,0,255,
192,0,0
1460 DATA 192,255,60,15,255,15,3,3,15,
255,60,255,255,252,252,252,192,0,0,255
,255,15,0,0,0,0,0,255,255
1470 DATA 255,252,240,0,0,0,255,255,63
,63,15,15,3,3,255,255,255,255,255,253,
245,213,255,255,255,255,87,87,87,87
1480 DATA 255,255,252,252,240,240,192,
192,85,105,105,85,255,255,255,255,87,9
5,127,255,255,255,255,255
1490 DATA 20,85,40,255,60,60,195,195,2
55,255,254,234,169,169,234,254,255,255
,191,171,90,90,171,191
1500 DATA 235,235,125,125,235,235,125,
125,0,0,255,63,15,15,15,15,0,0,240,252
,255,175,255,255
1510 DATA 85,85,85,85,85,85,85,85,0,0,
20,89,85,20,0,0,129,36,24,66,129,36,24
,66,0,0,0,60,12,255,255,60
1520 DATA 0,40,130,130,0,40,130,130
1530 DATA 0,1,2,3,4,4,5,1,6,7,8,9,3,9,
10,3,0,0,6,11,9,3,8,3,0,12,13,14,15,16
,6,17
1540 DATA 18,25,26,3,3,3,3,25,26,3,3,3
,3,25,26,3,3,3,3,25,26,3,3,3,3,25,26,3
,3,3,3,25,26,3,3,3,3,25,26,21
1550 DATA 0,18,3,19,20,3,3,19,20,3,3,1
9,20,3,3,19,20,3,3,19,20,3,3,19,20,3,3
,19,20,3,3,19,20,3,3,19,20,3,21,0
1560 DATA 0,0,18,22,23,3,3,22,23,3,3,2
2,23,3,3,22,23,3,3,22,23,3,3,22,23,3,3
,22,23,3,3,22,23,3,3,22,23,21,0,0
1570 DATA 42,43,83,123,124,125,167,127
,87,88,48,49,53,52,92,93,94,54,60,61,1
01,102,142,143,64,65,66,67,107,108
1580 DATA 70,71,111,110,150,151,73,74,
114,115,116,156,130,131,132,172,173,21
3,175,176,136,96,97,98,218,178,179
1590 DATA 180,220,260,222,223,263,303,
343,344,109,149,148,147,187,188,189,19
0,191,192,232,272,154,194,195,235
1600 DATA 236,237,211,210,209,249,289,
288,286,285,245,244,243,242,322,323,32
4,364,404,405,369,409,408,407,367,327
1610 DATA 251,291,331,371,411,412,293,
294,295,296,297,298,299,414,374,373,33
3,334,335,340,380,379,378,377,376
1620 DATA 422,423,424,425,426,386,427,
387,347,307,267,227,346,306,266,226,18
6,146,228,268,308,348,349,309,310
1630 DATA 311,271,170,269,229,313,273,
233,193,153,113,275,315,355,354,353,35
2,429,389,390,391,392,393,433,435
1640 DATA 436,437,397,396,356
1650 DATA 104,104,104,133,203,164,203,
162,0,189,247,6,145,207,200,232,224,8,
208,245,169,0,133,20,165,20,205,255,6
1660 DATA 208,249,164,203,169,0,170,14
5,207,200,232,224,8,208,248,96
1670 DATA 20,85,40,255,60,60,195,195
1680 DATA 104,169,7,160,11,162,144,32,
92,228,96,216,173,235,6,240,3,76,98,22
8,173,246,6,141,0,208,105,16
1690 DATA 141,1,208,173,120,2,201,11,2
08,16,56,173,246,6,233,1,201,22,144,3
1700 DATA 141,246,6,24,144,17,201,7,20
8,13,24,173,246,6,105,1,201,175,176,3,
141,246,6
```

```
1710 DATA 173,239,6,240,30,162,0,172,2
45,6,189,241,6,145,207,232,200,224,4,2
08,245,206,245,6,173,245,6
1720 DATA 208,27,141,239,6,24,144,21,1
73,132,2,208,16,169,200,141,245,6,173,
246,6,105,28,141,4,208,141,239,6,173,0

1730 DATA 208,41,8,240,5,169,1,141,240
,6,76,98,228
1740 DATA 0,0,1,1,1,1,3,3,5,5,9,9,17,1
7,240,240,16,16,16,48,48,16,16,16,16,1
6,16,24,24,20,20,18,18,255,255
1750 DATA 254,254,252,252,120,120,48,4
8,24,8,24,44,74,137,255,126,60
1760 DATA 104,169,7,160,61,162,155,32,
92,228,96,173,234,6,240,3,76,98,228,21
6,24,174,120,2,189,210,155,109
1770 DATA 237,6,141,237,6,24,189,170,1
55,109,236,6,141,236,6,162,0,172,236,6
,189,190,155,145,204,232,200,224,12
1780 DATA 208,245,173,237,6,141,3,208,
173,7,208,240,5,169,1,141,234,6,76,98,
228,0,8,62,28,28,8,28,42,8,20,54,0
1790 DATA 0,0,0,0,0,0,1,1,1,0,255,255,25
5,0,0,0,0,0,0,0,0,0,1,255,0,0,1,255,0,
0,1,255,0
```

# RAMSES' REVENGE

```
10 DATA 0,254,254,254,0,239,239,239
20 DATA 0,60,24,36,66,66,36,24
30 DATA 126,126,255,255,255,255,195,19
5
40 DATA 170,85,170,85,170,85,170,85
50 DATA 62,34,62,6,30,6,14,6
60 DATA 56,56,16,124,146,56,40,102
70 DATA 28,28,73,62,8,28,20,54
80 DATA 0,0,129,66,60,24,0,0
90 DATA 0,48,28,15,7,3,1,1
100 DATA 0,0,0,0,192,240,248,248
110 DATA 252,254,254,254,254,254,254,2
52
120 DATA 248,248,240,192,0,0,0,0
130 DATA 1,1,3,7,15,28,48,0
140 DATA 60,126,219,255,255,231,60,24
150 DATA 195,90,126,36,60,24,60,102
160 DATA 24,60,60,153,255,24,60,102
200 DIM CHAR$(16)
210 CHAR$="#$%&@()<>-=+*[]/"
220 CHSET=(PEEK(106)-8)*256:CHORG=5734
4
230 IF PEEK(CHSET+9)<>0 THEN 300
240 FOR I=0 TO 511:POKE CHSET+I,PEEK(C
HORG+I):NEXT I
250 FOR I=1 TO 16
260 CHPOS=CHSET+(ASC(CHAR$(I))-32)*8
270 FOR J=0 TO 7
280 READ A:POKE CHPOS+J,A
290 NEXT J:NEXT I
300 POKE 756,CHSET/256
400 ? "ß":SETCOLOR 2,0,0:POKE 752,1
410 ? "    5000 YEARS AGO, THE PHARAOH
      RAMSES WAS SWALLOWED UP BY THE
RED"
415 ? "SEA WHILE IN PURSUIT OF MOSES W
HO HAD FREED THE SLAVES OF EGYPT.  HOW
EVER,"
420 ? "BEFORE LEAVING THE GLORY OF EGY
PT,    RAMSES TRANSFERED ALL OF HIS PO
WER"
425 ? "TO THE GREAT GOLDEN KEYS, PRAYI
NG TO  THE EVIL GOD SETH THAT THEY MIG
HT"
430 ? "REMAIN SAFE; HIDDEN IN THE SACR
ED     PYRAMID."
440 ? "    ONLY NOW, HAVE YOU AND ANOTH
ER     EXPLORER STUMBLED ONTO THIS PRI
VIOUSLY"
445 ? "†UNDISCOVERED PYRAMID.  WILL ON
E OF     YOU RECOVER THE ALL POWERFUL G
OLDEN"
450 ? "KEYS, OR WILL BOTH OF YOU FALL
VICTIM TO THE CURSE OF RAMSES' REVENGE
?"
460 ? :? :? :? "          PRESS START TO
 BEGIN"
465 IF PEEK(53279)<>6 THEN 465
1000 POKE 559,0:? "ß":SETCOLOR 2,0,0
1010 POSITION 19,1:? "#":POSITION 18,2
:? "#####":POSITION 17,3:? "### ###":POSIT
ION 16,4:? "###   ###":POSITION 15,5:? "
##### [ #####"
1015 POSITION 19,3:? "@"
1020 POSITION 14,6:? "###          ###":POS
ITION 13,7:? "##### ####### #####":POSITION
 12,8:? "###             ###":POSITION 11,9
:? "######### ##### #########"
1030 POSITION 10,10:? "###
  ###":POSITION 9,11:? "##### ######### ######
### #####":POSITION 8,12:? "###          #####
          ###"
1035 POSITION 2,13:? "[":POSITION 36,1
3:? "["
1040 POSITION 7,13:? "############ ##### #####
 ############":POSITION 5,14:? "#####
                 #####"
1050 POSITION 0,15:? "&& &&########## ######
################ #########&& &&":POSITION 0,16
:? "&   &&&                          &
&&   &"
1060 POSITION 0,17:? "& &&&& &&&&&&&&
& &&& &&&&&&&&& &&&& &":POSITION 0,18
:? "& &&&&&&     &&&          &&&    &&&
&&& &"
1070 POSITION 0,19:? "&          &&
&&&&&&&     &&          &":POSITION 0,20
:? "&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
&&&&&"
1075 POSITION 14,8:? "$":POSITION 24,8
:? "$":POSITION 19,11:? "$":POSITION 1
9,13:? "$":POSITION 8,14:? "$":POSITIO
N 30,14:? "$"
1077 POSITION 6,17:? "$":POSITION 32,1
7:? "$"
1080 POSITION 32,2:? ">-":POSITION 33,
3:? "=":POSITION 32,4:? "*+":POSITION
2,14:? "%":POSITION 36,14:? "%"
1090 POSITION 12,21:? "RAMSES' REVENGE
"
1092 POKE 559,34
1995 A=602:B=636:R=0:R1=0:Z=3:Z1=3:MV=
19:BV=17:BV1=21:TT=1:MV1=19:SS=1:UU=1:
KE=0:KE1=0:N=1:N1=1
2000 SOUND 0,0,0,0:SOUND 1,0,0,0:SOUND
 2,0,0,0:SOUND 3,0,0,0
2001 SC=PEEK(88)+256*PEEK(89):C=PEEK(S
C+A-40):D=PEEK(SC+B-40):E=PEEK(SC+A+40
):F=PEEK(SC+B+40)
2005 G=PEEK(SC+A+1):H=PEEK(SC+B+1):I=P
EEK(SC+A-1):M=PEEK(SC+B-1):K=PEEK(SC+A
):L=PEEK(SC+B)
2010 J=STICK(0):J1=STICK(1):POKE SC+A,
9:POKE SC+B,8
2020 IF J=14 AND N AND C<>3 AND C<>59
 AND C<>6 AND C<>5 THEN POKE SC+A,0:A=A
-40:SOUND 0,150,10,6
2025 IF J1=14 AND N1 AND D<>3 AND D<>5
9 AND D<>6 AND D<>5 THEN POKE SC+B,0:B
=B-40:SOUND 0,200,10,6
2030 IF J=13 AND N AND E<>3 AND E<>59
 AND E<>6 AND E<>5 THEN POKE SC+A,0:A=A
+40:SOUND 0,150,10,6
2035 IF J1=13 AND N1 AND F<>3 AND F<>5
9 AND F<>6 AND F<>5 THEN POKE SC+B,0:B
=B+40:SOUND 0,200,10,6
2040 IF J=7 AND N AND G<>3 AND G<>59 A
ND G<>6 AND G<>5 THEN POKE SC+A,0:A=A+
1:SOUND 0,150,10,6
2045 IF J1=7 AND N1 AND H<>3 AND H<>59
 AND H<>6 AND H<>5 THEN POKE SC+B,0:B=
B+1:SOUND 0,200,10,6
2050 IF J=11 AND N AND I<>3 AND I<>59
 AND I<>6 AND I<>5 THEN POKE SC+A,0:A=A
-1:SOUND 0,150,10,6
2055 IF J1=11 AND N1 AND M<>3 AND M<>5
9 AND M<>6 AND M<>5 THEN POKE SC+B,0:B
=B-1:SOUND 0,200,10,6
2060 IF K=4 THEN R=R+1:SOUND 1,30,10,1
0:POSITION -1+R,21:? "$":IF R=3 THEN Z
=Z+1:R=0:POSITION 0,21:? "   "
2065 IF L=4 THEN R1=R1+1:SOUND 1,30,10
,10:POSITION 35+R1,21:? "$":IF R1=3 TH
EN Z1=Z1+1:R1=0:POSITION 36,21:? "   "

2070 POSITION 4,21:? "):";Z:IF Z=0 THE
N N=0:POSITION 2,15:? " "
2075 POSITION 32,21:? "(:";Z1:IF Z1=0
THEN N1=0:POSITION 36,15:? " "
2077 POSITION MV,6:? " ":MV=MV+TT:IF M
V=22 OR MV=16 THEN TT=-TT:SOUND 3,100,
12,8
2080 POSITION MV,6:? "]"
2085 POSITION MV1,10:? " ":MV1=MV1+SS:
IF MV1=12 OR MV1=26 THEN SS=-SS:SOUND
3,100,12,8
2087 POSITION MV1,10:? "/"
2090 POSITION BV,12:? " ":POSITION BV1
,12:? " ":BV=BV-UU:BV1=BV1+UU:IF BV=10
 OR BV=17 THEN UU=-UU:SOUND 3,100,12,8

2095 POSITION BV,12:? "(":POSITION BV1
,12:? "("
2100 IF K=61 OR K=15 OR K=28 THEN Z=Z-
1:SOUND 2,220,12,14:FOR W=1 TO 20:NEXT
 W:A=602
2105 IF L=61 OR L=15 OR L=28 THEN Z1=Z
1-1:SOUND 2,220,12,14:FOR W=1 TO 20:NE
XT W:B=636
2110 IF K=32 THEN KE=KE+1:GOSUB 2240:P
OKE SC+B,0:A=602:B=636:POSITION 19,3:?
 "@":POSITION -1+KE,22:? "@"
2115 IF L=32 THEN KE1=KE1+1:GOSUB 2240
:POKE SC+A,0:A=602:B=636:POSITION 19,3
:? "@":POSITION 28+KE1,22:? "@"
2117 IF KE=10 THEN 2210
2118 IF KE1=10 THEN 2220
2119 IF N=0 AND N1=0 THEN 2230
2200 GOTO 2000
2210 FOR Y=0 TO 3:SOUND Y,0,0,0:NEXT Y
:GRAPHICS 18:POSITION 4,5:? #6;"PLAYER
 1 WINS":GOTO 2231
2220 FOR Y=0 TO 3:SOUND Y,0,0,0:NEXT Y
:GRAPHICS 18:POSITION 4,5:? #6;"PLAYER
 2 WINS":GOTO 2231
2230 FOR Y=0 TO 3:SOUND Y,0,0,0:NEXT Y
:GRAPHICS 18:POSITION 4,5:? #6;"THE LE
GEND OF    THE CURSE IS TRUE!"
2231 FOR W=0 TO 0 STEP 0:NEXT W
2240 FOR Y=0 TO 3:SOUND Y,0,0,0:NEXT Y
:RESTORE 2255
2245 READ OO,PP:IF OO>0 THEN SOUND 0,O
O,10,8:FOR QQ=1 TO PP*1.3:NEXT QQ
2255 DATA 121,20,108,20,102,35,108,35,
121,35,121,20,108,20,102,20,81,20,102,
20,108,20,121,40,-1,0
2260 IF OO>0 THEN SOUND 0,0,0,0:FOR QQ
=1 TO 2:NEXT QQ:GOTO 2245
2265 RETURN
```

# ROBOT DUNGEON

```
10 REM ROBOT DUNGEON
20 REM BY ANSCHUETZ/WEISGERBER/ANSCHUE
TZ
30 REM (c) 1985, ANTIC PUBLISHING
40 ? "{CLEAR}WARNING! {INVERSE}ROBOT DUNGEON{NORMAL} WRITES
 TO"
50 ? "YOUR DISK!!!"
60 ? :? "Do you want instructions (Y/N
)?";
62 POKE 764,255:POKE 1540,15
65 IF (PEEK(764)=43 OR PEEK(764)=107)
THEN POKE 764,255:RUN "D:ROBOHELP"
67 IF (PEEK(764)=35 OR PEEK(764)=99) T
HEN POKE 764,255:GOTO 100
70 GOTO 65
100 GOTO 1060
110 GRAPHICS 2:POKE 53774,64:POKE 16,6
4:MEM=PEEK(106):DIM AA(4),BB(4),MM(4),
OA(4),OB(4),D$(12),M$(8):M$="    - :+"

120 CH=MEM-4:FP=(MEM-17)*256-1:SP=(MEM
-9)*256-1:POKE 53277,0:POKE 752,1:SETC
OLOR 3,0,0:IF RU THEN 240
130 FOR X=0 TO 511:POKE CH*256+X,PEEK(
57344+X):NEXT X:POKE 756,CH:? CHR$(125
)
140 ? "ENTER DEVICE NAME - CASSETTE US
ERS    SIMPLY USE C: - DISK USERS USE
SAME    FILENAME AS USED IN CREATOR"
150 TRAP 140:INPUT D$:IF D$(1,1)="C" T
HEN ? CHR$(125):GOTO 190
160 IF D$(1,1)<>"D" THEN ? CHR$(125):G
OTO 140
170 ? CHR$(125):? "READY DISK AND THEN
 PRESS {INVERSE}RETURN{NORMAL}      ON KEYBOARD":POKE 7
64,255:GOSUB 1600
180 OPEN #1,4,0,D$:GOTO 200
190 ? "PRESS {INVERSE}PLAY{NORMAL} ON RECORDER AND THEN
      PRESS {INVERSE}RETURN{NORMAL} ON KEYBOARD":POKE
764,255:OPEN #1,4,0,D$
200 TRAP 40000:FOR R=1 TO 20:FOR C=1 T
O 60:GET #1,A:POKE FP+(R-1)*60+C,A
210 NEXT C:NEXT R:FOR R=1 TO 20:FOR C=
1 TO 60:GET #1,A:POKE SP+(R-1)*60+C,A:
NEXT C:NEXT R
220 FOR X=8 TO 127:GET #1,A:POKE CH*25
6+X,A:NEXT X:GET #1,LYN:DIM A$(LYN):FO
R X=1 TO LYN:GET #1,A
230 A$(X,X)=CHR$(A):NEXT X:GET #1,STAT
:INPUT #1,SC:GET #1,R:GET #1,C:INPUT #
1,LI:GET #1,XA:GET #1,XB:CLOSE #1
240 CLOSE #1:IF STAT=0 THEN NXA=XA:NXB
=XB:NLI=LI:NR=R:NC=C:NSC=SC
250 POKE 559,0:FOR X=504 TO 511:READ A
:POKE X+CH*256,A:NEXT X:FOR X=256 TO 2
63:READ A:POKE X+CH*256,A:NEXT X
260 FOR X=224 TO 231:READ A:POKE X+CH*
256,A:NEXT X
270 GOSUB 1620:MM(0)=170:MM(1)=171:MM(
2)=172:MM(3)=173:MM(4)=174:GOSUB 430
280 FOR ZZ=0 TO 4:JW=1
290 IF MM(ZZ)=0 THEN JW=0:GOTO 360
300 OA(ZZ)=AA(ZZ):OB(ZZ)=BB(ZZ):SS=INT
(RND(0)*2)+1:ON SS GOTO 310,330
310 IF PEEK(1750)<(AA(ZZ)*8)+48 THEN A
A(ZZ)=AA(ZZ)-1*(AA(ZZ)>0):GOTO 350
320 POKE 53278,0:AA(ZZ)=AA(ZZ)+1*(AA(Z
Z)<20):GOTO 350
330 IF PEEK(1780)<(BB(ZZ)*16)+32 THEN
BB(ZZ)=BB(ZZ)-1*(BB(ZZ)>0):GOTO 350
340 BB(ZZ)=BB(ZZ)+1*(BB(ZZ)<10)
350 LOCATE AA(ZZ),BB(ZZ),CC:COLOR 1:IF
 CC<>0 THEN AA(ZZ)=OA(ZZ):BB(ZZ)=OB(ZZ
):JW=0
360 CR=PEEK(53252):ZX=PEEK(1750):ZY=PE
EK(1780):IF CR AND RA THEN 840
370 IF PEEK(1752) THEN GOSUB 1110
380 IF PEEK(53253) THEN POKE 53278,0
390 IF JW=1 THEN POSITION OA(ZZ),OB(ZZ
):? #6;CHR$(0):POSITION AA(ZZ),BB(ZZ):
? #6;CHR$(MM(ZZ))
400 IF PEEK(764)<255 THEN POKE 764,255
:GOTO 820
410 IF PEEK(53279)>0 AND PEEK(53279)<7
 THEN GOSUB 1500
420 NEXT ZZ:RA=1:POKE 54286,192:SOUND
0,0,0,0:GOTO 280
430 POKE 1752,0:POKE 559,0:COLOR 33:PL
OT 1,0:DRAWTO 18,0:DRAWTO 18,8:DRAWTO
1,8:DRAWTO 1,0
440 MM(0)=170:MM(1)=171:MM(2)=172:MM(3
)=173:MM(4)=174
450 COLOR 0:FOR Y=1 TO 7:PLOT 2,Y:DRAW
TO 17,Y:NEXT Y
460 X=PEEK((R-1)*60+C+FP):NX=X:IF X>12
7 THEN POSITION 15,6:? #6;CHR$(15):X=X
-128
470 COLOR 130:IF X>63 THEN PLOT 13,0:D
RAWTO 15,0:X=X-64
480 IF X>31 THEN PLOT 4,0:DRAWTO 6,0:X
=X-32
490 COLOR 131:IF X>15 THEN PLOT 18,3:D
RAWTO 18,5:X=X-16
500 COLOR 130:IF X>3 THEN PLOT 13,8:DR
AWTO 15,8:X=X-4
510 IF X>1 THEN PLOT 4,8:DRAWTO 6,8:X=
X-2
520 COLOR 131:IF X>0 THEN PLOT 1,3:DRA
WTO 1,5
530 XX=PEEK((R-1)*60+C+SP):NXX=XX
540 COLOR 33:IF XX>127 THEN PLOT 5,4:D
RAWTO 7,4:PLOT 12,4:DRAWTO 14,4:XX=XX-
128
550 IF XX>63 THEN COLOR 7:PLOT 9,6:XX=
XX-64
560 IF XX>31 THEN COLOR 5:PLOT 13,2:XX
=XX-32
570 IF XX>15 THEN COLOR 6:PLOT 9,4:DRA
WTO 10,4:XX=XX-16
580 IF XX>7 THEN COLOR 136:PLOT 9,2:CO
LOR 137:PLOT 10,2:XX=XX-8
590 IF XX>3 THEN COLOR 4:PLOT 6,6:XX=X
X-4
600 COLOR 33:IF XX>1 THEN PLOT 12,1:DR
AWTO 12,4:XX=XX-2
610 IF XX>0 THEN PLOT 7,7:DRAWTO 7,4
620 POKE 53278,0:FOR QW=0 TO 4
630 AA(QW)=INT(RND(0)*12)+4:BB(QW)=INT
(RND(0)*3)+3:LOCATE AA(QW),BB(QW),CC:I
F CC<>0 THEN 630
640 POSITION AA(QW),BB(QW):? #6;CHR$(M
M(QW)):NEXT QW:FOR X=1 TO (C>40)+(C>20
)+1
650 QQ=INT(RND(0)*5):IF MM(QQ)>100 THE
N MM(QQ)=MM(QQ)-128:POSITION AA(QQ),BB
(QQ):? #6;CHR$(MM(QQ)):NEXT X:GOTO 670

660 X=X-1:NEXT X
670 SETCOLOR 0,NX+8,8:SETCOLOR 4,NXX+1
4,6:RA=0:CC=(C+760)*(C>40)+(C+380)*(C>
20 AND C<41)+C*(C<21):RN=(R-1)*20+CC
680 IF RN=400 OR RN=800 THEN POSITION
17,1:? #6;CHR$(224)
690 IF RN=1200 THEN POSITION 17,1:? #6
;CHR$(255)
700 POKE 656,1:POKE 657,28:? SC;:POKE
656,2:POKE 657,14:? RN;"   ";:POKE 657
,33:? LI;" ";
710 POKE 756,CH:POKE 1756,0:IF LI=0 TH
EN 730
720 POKE 77,0:POKE 1755,39:POKE 53248,
0:POKE 764,255:POKE 1750,XA:POKE 1780,
XB:POKE 53248,XA:POKE 559,62:RETURN
730 POP :POKE 54286,64:POKE 53765,192:
POKE 559,62:POKE 1780,XB:POKE 53248,XA
:IF PEEK(53279)=6 THEN 750
740 GOTO 730
750 ? CHR$(125):OPEN #1,4,0,"K":POKE 7
64,255
760 ? "DO YOU WANT TO {INVERSE}R{NORMAL}EPLAY THIS SAME
      DUNGEON OR {INVERSE}L{NORMAL}OAD IN ANOTHER ONE?
";:GET #1,A
770 IF A=76 THEN POKE 53277,0:GRAPHICS
 0:RUN
780 IF A=82 AND STAT=1 THEN RU=1:GOTO
810
790 IF A=82 THEN XA=NXA:XB=NXB:LI=NLI:
R=NR:C=NC:SC=NSC:RU=1:GOTO 810
800 POKE 764,255:? CHR$(125):GOTO 760
810 POKE 53248,0:POKE 53249,0:RESTORE
:GOTO 120
820 POKE 54286,64:POKE 53765,192:IF PE
EK(764)<255 THEN POKE 764,255:POKE 542
86,192:GOTO 410
830 GOTO 820
840 IF CR=1 OR CR=4 OR CR=5 OR CR=9 TH
EN POKE 54286,64:POKE 53765,192:POKE 5
59,0:GOSUB 1050:GOTO 970
850 IF CR=8 AND ZX>115 AND ZX<137 AND
ZY>49 AND ZY<81 THEN GOSUB 1020:GOTO 9
70
860 IF CR=8 AND ZX>177 AND ZY<66 AND R
N=1200 THEN GOSUB 1050:LI=LI+1:GOTO 13
50
870 IF CR=8 AND ZX>177 AND ZY<66 THEN
R=1:C=C+1:GOSUB 1050:LI=LI+1:GOSUB 124
0:XA=89:XB=145:GOSUB 430:GOTO 280
880 IF CR=8 THEN POKE 559,0:POKE 54286
,64:POKE 53249,0:POKE 207,0:GOTO 1160
890 GH=(R-1)*60+C+SP:NSP=GH-2048
900 IF CR=2 AND ZX<104 THEN SC=SC+250:
POSITION 6,6:? #6;CHR$(0):GOSUB 990:PO
KE GH,PEEK(GH)-4:GOTO 980
910 IF CR=2 AND ZX>145 AND ZY<99 THEN
SC=SC+1000:POSITION 13,2:? #6;CHR$(0):
GOSUB 990:POKE GH,PEEK(GH)-32:GOTO 980

920 IF CR=2 AND ZX>161 AND ZX<176 AND
ZY>115 THEN LI=LI+5:GOSUB 1040:POKE NS
P,PEEK(NSP)-128:GOTO 980
```

```
930 IF CR=2 AND ZY>110 THEN POSITION 9
,6:? #6;CHR$(0):LI=LI+1:GOSUB 1000:POK
E GH,PEEK(GH)-64:GOTO 980
940 IF CR=2 AND ZY<110 THEN POSITION 9
,4:? #6;"  ":GOSUB 1010:FOR QC=0 TO 4:
MM(QC)=0:F=1
950 IF F THEN F=0:POSITION AA(QC),BB(Q
C):? #6;" ":NEXT QC:POKE GH,PEEK(GH)-1
6
960 GOTO 980
970 POKE 1750,XA:POKE 1780,XB:GOSUB 43
0:POKE 53278,0:GOTO 370
980 POKE 53278,0:GOTO 370
990 SOUND 0,25,10,8:DE=1^1:SOUND 0,0,0
,0:POKE 656,1:POKE 657,28:? SC;:RETURN

1000 SOUND 0,200,10,8:DE=1^1:SOUND 0,0
,0,0:POKE 656,2:POKE 657,33:? LI;:RETU
RN
1010 SOUND 0,150,2,8:DE=1^1^1:SOUND 0,
0,0,0:SC=SC+500:POKE 656,1:POKE 657,28
:? SC;:RETURN
1020 XA=89:XB=145:LI=LI-1:SOUND 0,100,
2,8:SOUND 1,101,2,8:DE=1^1^1^1:SOUND 0
,0,0,0:SOUND 1,0,0,0
1030 CC=C:R=1:C=(CC<21)+21*(CC>20 AND
CC<41)+41*(CC>40):RETURN
1040 POSITION 15,6:? #6;CHR$(0):SOUND
0,100,10,8:DE=1^1:SOUND 0,0,0,0:POKE 6
56,2:POKE 657,33:? LI;:RETURN
1050 POKE 54286,64:POKE 53248,0:POKE 1
780,0:POKE 207,0:LI=LI-1:POKE 53278,0:
POKE 53249,0:RETURN
1060 GRAPHICS 1:POKE 53774,64:POKE 16,
64:TR=PEEK(710):SETCOLOR 2,0,0:DL=PEEK
(560)+256*PEEK(561)
1070 POKE DL+14,23:POKE DL+18,2:POKE 5
4276,4:SETCOLOR 0,8,4:POSITION 5,9:? #
6;"ROBOT DUNGEON"
1080 POSITION 4,11:? #6;"press start t
o begin":POSITION 9,13:? #6;"ANSCHUETZ
/WEISGERBER/ANSCHUETZ"
1090 IF PEEK(53279)=6 THEN POKE 710,TR
:GOTO 110
1100 GOTO 1090
1110 WX=INT((PEEK(1753)-45)/8)
1120 WY=INT((PEEK(1754)-27)/16)
1130 LOCATE WX,WY,WZ:FOR X=0 TO 4:IF W
Z=MM(X) THEN MM(X)=0:POSITION AA(X),BB
(X):? #6;CHR$(0):SC=SC+10:GOSUB 1150
1140 NEXT X:POKE 656,1:POKE 657,28:? S
C;:POKE 53278,0:POKE 1752,0:RETURN
1150 SOUND 0,200,12,10:DE=1^1:SOUND 0,
0,0,0:RETURN
1160 POKE 1756,0:POKE 53765,192:YX=PEE
K(1750):YY=PEEK(1780):POKE 559,0
1170 IF YX<64 THEN C=C-1:XA=184:XB=97:
GOSUB 430:GOTO 280
1180 IF YX>184 THEN C=C+1:XA=64:XB=97:
GOSUB 430:GOTO 280
1190 IF YX<128 THEN 1220
1200 IF YY<128 THEN C=C+(R/2<>INT(R/2)
):R=R+1:XA=89:XB=149:GOSUB 430:GOTO 28
0
1210 C=C+(R/2<>INT(R/2)):R=R-1:XA=89:X
B=47:GOSUB 430:GOTO 280
1220 IF YY<128 THEN C=C-(R/2=INT(R/2))
:R=R+1:XA=161:XB=149:GOSUB 430:GOTO 28
0
1230 C=C-(R/2=INT(R/2)):R=R-1:XA=161:X
B=47:GOSUB 430:GOTO 280
1240 POKE 54286,64:POKE 53765,192:POKE
 559,0
1250 COLOR 33:FOR SY=0 TO 9:PLOT 0,SY:
DRAWTO 19,SY:NEXT SY:COLOR 0:PLOT 0,0:
DRAWTO 5,0:DRAWTO 11,6
1260 DRAWTO 19,6:PLOT 0,1:DRAWTO 5,1:D
RAWTO 11,7:DRAWTO 19,7:PLOT 0,2:DRAWTO
 5,2:DRAWTO 10,7:PLOT 12,4
1270 DRAWTO 19,4:PLOT 12,5:DRAWTO 19,5
:COLOR 131:PLOT 11,6:DRAWTO 11,7
1280 POKE 1780,66:POKE 559,62:FOR PX=4
8 TO 88 STEP 8:POKE 53248,PX:GOSUB 147
0:FOR W=1 TO 75:NEXT W:NEXT PX
1290 FOR PY=82 TO 146 STEP 16:POKE 178
0,PY:POKE 53248,PX:GOSUB 1470:PX=PX+8:
FOR W=1 TO 75:NEXT W:NEXT PY
1300 FOR SX=31 TO 24 STEP -1:POKE CH*2
56+SX,0:FOR W=1 TO 15:NEXT W:SOUND 0,5
X+50,10,8:NEXT SX:SOUND 0,0,0,0
1310 POKE 53248,PX:GOSUB 1470:FOR W=1
TO 100:NEXT W:POKE 53248,PX+8:GOSUB 14
70
1320 FOR SX=24 TO 31:POKE CH*256+SX,56
:FOR W=1 TO 15:NEXT W:SOUND 0,SX+50,10
,8:NEXT SX:SOUND 0,0,0,0
1330 FOR PX=152 TO 200 STEP 8:POKE 532
48,PX:GOSUB 1470:FOR W=1 TO 75:NEXT W:
NEXT PX
1340 POKE 559,0:COLOR 0:PLOT 0,0:DRAWT
O 0,9:DRAWTO 19,9:DRAWTO 19,0:RETURN
1350 POKE 54286,64:POKE 53765,192:POKE
 559,0:MP=14
1360 COLOR 33:FOR SY=0 TO 9:PLOT 0,SY:
DRAWTO 19,SY:NEXT SY:COLOR 0:FOR SY=0
TO 7:PLOT 12,SY:DRAWTO 19,SY:NEXT SY
1370 FOR SY=4 TO 7:PLOT 0,SY:DRAWTO 7,
SY:NEXT SY:COLOR 131:FOR SY=6 TO 7:PLO
T 8,SY:DRAWTO 11,SY:NEXT SY
1380 POSITION INT((20-LEN(A$))/2),9:?
#6;A$
1390 POKE 1780,146:POKE 559,62:FOR PX=
48 TO 104 STEP 8:POKE 53248,PX:GOSUB 1
470:FOR W=1 TO 75:NEXT W:NEXT PX
1400 FOR SX=31 TO 24 STEP -1:POKE CH*2
56+SX,0:FOR W=1 TO 15:NEXT W:SOUND 0,5
X+50,10,8:NEXT SX:SOUND 0,0,0,0
1410 FOR PX=112 TO 144 STEP 8:POKE 532
48,PX:GOSUB 1470:FOR W=1 TO 75:NEXT W:
NEXT PX
1420 FOR SX=24 TO 31:POKE CH*256+SX,56
:FOR W=1 TO 15:NEXT W:SOUND 0,SX+50,10
,8:NEXT SX:SOUND 0,0,0,0
1430 POKE 53248,152:GOSUB 1470
1440 POSITION 14,1:? #6;"YOU":POSITION
 14,2:? #6;"MADE":POSITION 14,3:? #6;"
IT<<<"
1450 FOR EX=1 TO 4:POKE 1780,139:FOR W
=1 TO 10:NEXT W:POKE 1780,146:GOSUB 14
70:FOR W=1 TO 10:NEXT W:NEXT EX
1460 GOTO 1460
1470 SOUND 0,66,12,8:FOR WN=1 TO 15:NE
XT WN:SOUND 0,0,0,0
1480 MP=MP-1:IF MP<9 AND MP>0 THEN POS
ITION 0,7:? #6;M$(MP,8)
1490 RETURN
1500 POKE 54286,64:POKE 53765,192:? CH
R$(125)
1510 IF D$(1,1)="D" THEN 1530
1520 ? "PRESS RECORD AND PLAY ON RECOR
DER     AND THEN PRESS RETURN ON KEYBO
ARD":GOTO 1540
1530 ? "READY DISK AND THEN PRESS RETU
RN      ON KEYBOARD":POKE 764,255:GOSU
B 1600
1540 POKE 764,255:OPEN #1,8,0,D$:FOR N
R=1 TO 20:FOR NC=1 TO 60:NA=PEEK(FP+(N
R-1)*60+NC):PUT #1,NA:NEXT NC
1550 NEXT NR:FOR NR=1 TO 20:FOR NC=1 T
O 60:NA=PEEK(SP+(NR-1)*60+NC):PUT #1,N
A:NEXT NC:NEXT NR
1560 FOR P=8 TO 127:NA=PEEK(CH*256+P):
PUT #1,NA:NEXT P
1570 PUT #1,LEN(A$):FOR P=1 TO LEN(A$)
:PUT #1,ASC(A$(P,P)):NEXT P:PUT #1,0
1580 PRINT #1,SC;PUT #1,R:PUT #1,C:PRI
NT #1,LI:PUT #1,XA:PUT #1,XB:CLOSE #1:
GOSUB 1730:POKE 54286,192
1590 RETURN
1600 IF PEEK(764)<255 THEN RETURN
1610 GOTO 1600
1620 PM=(MEM-12)*256:FOR X=PM TO PM+13
:READ A:POKE X,A:NEXT X:FOR X=PM+256 T
O PM+261:READ A:POKE X,A:NEXT X
1630 FOR X=1536 TO 1706:READ A:POKE X,
A:NEXT X:FOR X=(MEM-10)*256 TO (MEM-10
)*256+248:READ A:POKE X,A
1640 IF A=86 THEN POKE X,MEM-10
1650 NEXT X:FOR X=(MEM-2)*256 TO (MEM-
2)*256+68:READ A:POKE X,A
1660 IF A=86 THEN POKE X,MEM-10
1670 NEXT X:POKE (MEM-10)*256+243,MEM-
2
1680 FOR X=207 TO 209:POKE X,0:NEXT X:
POKE 623,1:POKE 1788,PM/256:POKE 53277
,3:POKE 54279,PM/256-4
1690 POKE 1784,14:POKE 1785,6:POKE 175
0,89:POKE 1780,145:POKE 704,52:POKE 70
5,194:POKE 1752,0:X=USR(1696)
1700 FOR X=PM TO PM+13:POKE X,0:NEXT X

1710 DL=PEEK(560)+256*PEEK(561):POKE D
L+15,194:POKE DL+18,130:POKE DL+19,130
:POKE DL+20,130
1720 POKE 512,111:POKE 513,MEM-10:POKE
 752,1:POKE 1756,1:IF STAT THEN XA=89:
XB=145:LI=10:R=1:C=1:SC=0
1730 ? CHR$(125):TJ=PEEK(660)+256*PEEK
(661):FOR EA=0 TO 39:POKE EA+TJ,1:POKE
 EA+TJ+120,1:NEXT EA
1740 FOR EA=40 TO 80 STEP 40:POKE TJ+E
A,1:POKE TJ+EA+39,1:NEXT EA
1750 POKE 656,1:POKE 657,2:? A$;:POKE
656,1:POKE 657,21:? "SCORE: ";SC;
1760 POKE 656,2:POKE 657,2:? "ROOM NUM
BER ";RN;" ";:POKE 656,2:POKE 657,21:?
 "HIT POINTS: ";LI;:RETURN
1770 DATA 0,0,0,224,160,191,170,234,3,
3,15,15,63,63,255,255,0,24,24,24,24,0,
24,0
```

```
1780 DATA 0,56,254,84,40,56,146,124,56
,56,56,40,40,108,0,0,0,0,0,16
1790 DATA 162,3,189,244,6,240,89,56,22
1,240,6,240,83,141,254,6,106,141,255,6
,142,253,6,24,169,0,109,253,6,24,109
1800 DATA 252,6,133,204,133,206,189,24
0,6,133,203,173,254,6,133,205,189,248,
6,170,232,46,255,6,144,16,168,177
1810 DATA 203,145,205,169,0,145,203,13
6,202,208,244,76,87,6,160,0,177,203,14
5,205,169,0,145,203,200,202,208,244
1820 DATA 174,253,6,173,254,6,157,240,
6,189,236,6,240,48,133,203,24,138,141,
253,6,109,235,6,133,204,24,173,253
1830 DATA 6,109,252,6,133,206,189,240,
6,133,205,189,248,6,170,160,0,177,203,
145,205,200,202,208,248,174,253,6,169
1840 DATA 0,157,236,6,202,48,3,76,2,6,
76,98,228,0,0,104,169,7,162,6,160,0,32
,92,228,96
1850 DATA 72,138,72,165,207,201,1,240,
34,173,132,2,201,1,240,51,173,120,2,20
1,15,240,44,173,214,6,133,209,173,244
1860 DATA 6,141,245,6,173,120,2,133,20
8,169,1,133,207,216,24,166,208,165,209
,125,79,86,133,209,141,1,208,24,173
1870 DATA 245,6,125,95,86,141,245,6,14
1,10,212,169,111,141,0,2,104,170,104,6
4,0,0,0,0,0,0,1,1,1,0,255,255,255
1880 DATA 0,0,0,0,0,0,0,0,0,1,255,0,0,
1,255,0,0,1,255,0
1890 DATA 72,138,72,173,215,6,201,1,24
0,35,169,1,141,215,6,216,24,174,120,2,
189,79,86,109,214,6,141,214,6
1900 DATA 141,0,208,24,189,95,86,109,2
44,6,141,244,6,24,144,5,169,0,141,215,
6,141,10,212,169,173,141,0,2
1910 DATA 104,170,104,64,72,173,5,208,
201,0,240,48,201,4,240,15,169,0,234,23
4,141,1,208,133,207,234,234,234
1920 DATA 24,144,29,201,4,208,25,165,2
09,141,217,6,173,245,6,141,218,6,169,0
,234,234,141,1,208,133,207,169,1
1930 DATA 141,216,6,169,0,234,234,234,
141,10,212,169,0,141,0,2,169,94,141,1,
2,104,64
1940 DATA 72,173,220,6,201,1,240,46,21
6,173,219,6,24,105,1,141,219,6,201,60,
208,13,169,82,141,4,210,169,193,141,5
1950 DATA 210,24,144,19,201,120,208,15
,169,90,141,4,210,169,193,141,5,210,16
9,0,141,219,6,141,10,212,169,0,141,0,2

1960 DATA 169,86,141,1,2,104,64
```

# ROBOT DUNGEON – 2017

```
2 REM MODIFIED ON 7/31/2017 BY ERIC AN
SCHUETZ TO ALLOW VARIABLE SIZE OF MAZE
 FROM 5 TO 20
5 REM MODIFIED CODE HAS VARIABLES SIZE
, MAXSIZE, AND CHEAT
10 REM ROBOT DUNGEON
20 REM BY ANSCHUETZ/WEISGERBER/ANSCHUE
TZ
30 REM (C) 1985, ANTIC PUBLISHING
40 ? "{CLR}WARNING! {INV}ROBOT DUNGEON{NOR} WRITES
 TO"
50 ? "YOUR DISK!!!"
60 ? :? "Do you want instructions (Y/N
)?";
62 POKE 764,255:POKE 1540,15
65 IF (PEEK(764)=43 OR PEEK(764)=107)
THEN POKE 764,255:RUN "D:ROBOHELP"
67 IF (PEEK(764)=35 OR PEEK(764)=99) T
HEN POKE 764,255:GOTO 100
70 GOTO 65
100 MAXSIZE=20:GOTO 1060
110 GRAPHICS 2:POKE 53774,64:POKE 16,6
4:MEM=PEEK(106):DIM AA(4),BB(4),MM(4),
OA(4),OB(4),D$(12),M$(8):M$="    {INV}+{NOR}"

120 CH=MEM-4:FP=(MEM-17)*256-1:SP=(MEM
-9)*256-1:POKE 53277,0:POKE 752,1:SETC
OLOR 3,0,0:IF RU THEN 240
130 FOR X=0 TO 511:POKE CH*256+X,PEEK(
57344+X):NEXT X:POKE 756,CH:? CHR$(125
)
135 CHEAT=0:? CHR$(125):? "WALK THROUG
H WALLS AND MONSTERS? (Y/N)"
137 TRAP 135:INPUT D$:IF D$(1,1)="Y" T
HEN CHEAT=1
138 ? CHR$(125)
140 ? "ENTER DEVICE NAME - CASSETTE US
ERS    SIMPLY USE C: - DISK USERS USE
SAME    FILENAME AS USED IN CREATOR"
150 TRAP 140:INPUT D$:IF D$(1,1)="C" T
HEN ? CHR$(125):GOTO 190
160 IF D$(1,1)<>"D" THEN ? CHR$(125):G
OTO 140
170 ? CHR$(125):? "READY DISK AND THEN
 PRESS {INV}RETURN{NOR}      ON KEYBOARD":POKE 7
64,255:GOSUB 1600
180 OPEN #1,4,0,D$:GOTO 200
190 ? "PRESS {INV}PLAY{NOR} ON RECORDER AND THEN
       PRESS {INV}RETURN{NOR} ON KEYBOARD":POKE
764,255:OPEN #1,4,0,D$
200 TRAP 40000:INPUT #1,SIZE:IF SIZE<5
 OR SIZE>MAXSIZE THEN ? "ERROR: SIZE N
OT BETWEEN 5 AND 20":END
205 FOR R=1 TO SIZE:FOR C=1 TO SIZE*3:
GET #1,A:POKE FP+(R-1)*SIZE*3+C,A:NEXT
 C:NEXT R
210 FOR R=1 TO SIZE:FOR C=1 TO SIZE*3:
GET #1,A:POKE SP+(R-1)*SIZE*3+C,A:NEXT
 C:NEXT R
220 FOR X=8 TO 127:GET #1,A:POKE CH*25
6+X,A:NEXT X:GET #1,LYN:DIM A$(LYN):FO
R X=1 TO LYN:GET #1,A
230 A$(X,X)=CHR$(A):NEXT X:GET #1,STAT
:INPUT #1,SC:GET #1,R:GET #1,C:INPUT #
1,LI:GET #1,XA:GET #1,XB:CLOSE #1
240 CLOSE #1:IF STAT=0 THEN NXA=XA:NXB
=XB:NLI=LI:NR=R:NC=C:NSC=SC
250 POKE 559,0:FOR X=504 TO 511:READ A
:POKE X+CH*256,A:NEXT X:FOR X=256 TO 2
63:READ A:POKE X+CH*256,A:NEXT X
260 FOR X=224 TO 231:READ A:POKE X+CH*
256,A:NEXT X
270 GOSUB 1620:MM(0)=170:MM(1)=171:MM(
2)=172:MM(3)=173:MM(4)=174:GOSUB 430
280 FOR ZZ=0 TO 4:JW=1
290 IF MM(ZZ)=0 THEN JW=0:GOTO 360
300 OA(ZZ)=AA(ZZ):OB(ZZ)=BB(ZZ):SS=INT
(RND(0)*2)+1:ON SS GOTO 310,330
310 IF PEEK(1750)<(AA(ZZ)*8)+48 THEN A
A(ZZ)=AA(ZZ)-1*(AA(ZZ)>0):GOTO 350
320 POKE 53278,0:AA(ZZ)=AA(ZZ)+1*(AA(Z
Z)<20):GOTO 350
330 IF PEEK(1780)<(BB(ZZ)*16)+32 THEN
BB(ZZ)=BB(ZZ)-1*(BB(ZZ)>0):GOTO 350
340 BB(ZZ)=BB(ZZ)+1*(BB(ZZ)<10)
350 LOCATE AA(ZZ),BB(ZZ),CC:COLOR 1:IF
 CC<>0 THEN AA(ZZ)=OA(ZZ):BB(ZZ)=OB(ZZ
):JW=0
360 CR=PEEK(53252):ZX=PEEK(1750):ZY=PE
EK(1780):IF CR AND RA THEN 840
370 IF PEEK(1752) THEN GOSUB 1110
380 IF PEEK(53253) THEN POKE 53278,0
390 IF JW=1 THEN POSITION OA(ZZ),OB(ZZ
):? #6;CHR$(0):POSITION AA(ZZ),BB(ZZ):
? #6;CHR$(MM(ZZ))
400 IF PEEK(764)<255 THEN POKE 764,255
:GOTO 820
410 IF PEEK(53279)>0 AND PEEK(53279)<7
 THEN GOSUB 1500
420 NEXT ZZ:RA=1:POKE 54286,192:SOUND
0,0,0,0:GOTO 280
430 POKE 1752,0:POKE 559,0:COLOR 33:PL
OT 1,0:DRAWTO 18,0:DRAWTO 18,8:DRAWTO
1,8:DRAWTO 1,0
440 MM(0)=170:MM(1)=171:MM(2)=172:MM(3
)=173:MM(4)=174
450 COLOR 0:FOR Y=1 TO 7:PLOT 2,Y:DRAW
TO 17,Y:NEXT Y
460 X=PEEK((R-1)*SIZE*3+C+FP):NX=X:IF
X>127 THEN POSITION 15,6:? #6;CHR$(15)
:X=X-128
470 COLOR 130:IF X>63 THEN PLOT 13,0:D
RAWTO 15,0:X=X-64
480 IF X>31 THEN PLOT 4,0:DRAWTO 6,0:X
=X-32
490 COLOR 131:IF X>15 THEN PLOT 18,3:D
RAWTO 18,5:X=X-16
500 COLOR 130:IF X>3 THEN PLOT 13,8:DR
AWTO 15,8:X=X-4
510 IF X>1 THEN PLOT 4,8:DRAWTO 6,8:X=
X-2
520 COLOR 131:IF X>0 THEN PLOT 1,3:DRA
WTO 1,5
530 XX=PEEK((R-1)*SIZE*3+C+SP):NXX=XX
540 COLOR 33:IF XX>127 THEN PLOT 5,4:D
RAWTO 7,4:PLOT 12,4:DRAWTO 14,4:XX=XX-
128
550 IF XX>63 THEN COLOR 7:PLOT 9,6:XX=
XX-64
560 IF XX>31 THEN COLOR 5:PLOT 13,2:XX
=XX-32
570 IF XX>15 THEN COLOR 6:PLOT 9,4:DRA
WTO 10,4:XX=XX-16
580 IF XX>7 THEN COLOR 136:PLOT 9,2:CO
LOR 137:PLOT 10,2:XX=XX-8
590 IF XX>3 THEN COLOR 4:PLOT 6,6:XX=X
X-4
600 COLOR 33:IF XX>1 THEN PLOT 12,1:DR
AWTO 12,4:XX=XX-2
610 IF XX>0 THEN PLOT 7,7:DRAWTO 7,4
620 POKE 53278,0:FOR QW=0 TO 4
630 AA(QW)=INT(RND(0)*12)+4:BB(QW)=INT
(RND(0)*3)+3:LOCATE AA(QW),BB(QW),CC:I
F CC<>0 THEN 630
640 POSITION AA(QW),BB(QW):? #6;CHR$(M
M(QW)):NEXT QW:FOR X=1 TO (C>SIZE*2)+(
C>SIZE)+1
650 QQ=INT(RND(0)*5):IF MM(QQ)>100 THE
N MM(QQ)=MM(QQ)-128:POSITION AA(QQ),BB
(QQ):? #6;CHR$(MM(QQ)):NEXT X:GOTO 670

660 X=X-1:NEXT X
670 SETCOLOR 0,NX+8,8:SETCOLOR 4,NXX+1
4,6:RA=0
675 CC=(C+2*SIZE*(SIZE-1))*(C>SIZE*2)+
(C+SIZE*(SIZE-1))*(C>SIZE AND C<(SIZE*
2+1))+C*(C<(SIZE+1))
677 RN=(R-1)*SIZE+CC
680 IF RN=SIZE*SIZE OR RN=2*SIZE*SIZE
THEN POSITION 17,1:? #6;CHR$(224)
690 IF RN=3*SIZE*SIZE THEN POSITION 17
,1:? #6;CHR$(255)
700 POKE 656,1:POKE 657,28:? SC;:POKE
656,2:POKE 657,14:? RN;"   ";:POKE 657
,33:? LI;" ";
710 POKE 756,CH:POKE 1756,0:IF LI=0 TH
EN 730
720 POKE 77,0:POKE 1755,39:POKE 53248,
0:POKE 764,255:POKE 1750,XA:POKE 1780,
XB:POKE 53248,XA:POKE 559,62:RETURN
730 POP :POKE 54286,64:POKE 53765,192:
POKE 559,62:POKE 1780,XB:POKE 53248,XA
:IF PEEK(53279)=6 THEN 750
740 GOTO 730
750 ? CHR$(125):OPEN #1,4,0,"K":POKE 7
64,255
760 ? "DO YOU WANT TO {INV}R{NOR}EPLAY THIS SAME
     DUNGEON OR {INV}L{NOR}OAD IN ANOTHER ONE?
":GET #1,A
770 IF A=76 THEN POKE 53277,0:GRAPHICS
 0:RUN
780 IF A=82 AND STAT=1 THEN RU=1:GOTO
810
790 IF A=82 THEN XA=NXA:XB=NXB:LI=NLI:
R=NR:C=NC:SC=NSC:RU=1:GOTO 810
800 POKE 764,255:? CHR$(125):GOTO 760
810 POKE 53248,0:POKE 53249,0:RESTORE
:GOTO 120
820 POKE 54286,64:POKE 53765,192:IF PE
EK(764)<255 THEN POKE 764,255:POKE 542
86,192:GOTO 410
830 GOTO 820
840 IF CHEAT THEN GOTO 850
845 IF CR=1 OR CR=4 OR CR=5 OR CR=9 TH
EN POKE 54286,64:POKE 53765,192:POKE 5
59,0:GOSUB 1050:GOTO 970
850 IF CR=8 AND ZX>115 AND ZX<137 AND
ZY>49 AND ZY<81 THEN GOSUB 1020:GOTO 9
70
860 IF CR=8 AND ZX>177 AND ZY<66 AND R
```

```
N=3*SIZE*SIZE THEN GOSUB 1050:LI=LI+1:
GOTO 1350
870 IF CR=8 AND ZX>177 AND ZY<66 THEN
R=1:C=C+1:GOSUB 1050:LI=LI+1:GOSUB 124
0:XA=89:XB=145:GOSUB 430:GOTO 280
880 IF CR=8 THEN POKE 559,0:POKE 54286
,64:POKE 53249,0:POKE 207,0:GOTO 1160
890 GH=(R-1)*(SIZE*3)+C+SP:NSP=GH-2048

900 IF CR=2 AND ZX<104 THEN SC=SC+250:
POSITION 6,6:? #6;CHR$(0):GOSUB 990:PO
KE GH,PEEK(GH)-4:GOTO 980
910 IF CR=2 AND ZX>145 AND ZY<99 THEN
SC=SC+1000:POSITION 13,2:? #6;CHR$(0):
GOSUB 990:POKE GH,PEEK(GH)-32:GOTO 980

920 IF CR=2 AND ZX>161 AND ZX<176 AND
ZY>115 THEN LI=LI+5:GOSUB 1040:POKE NS
P,PEEK(NSP)-128:GOTO 980
930 IF CR=2 AND ZY>110 THEN POSITION 9
,6:? #6;CHR$(0):LI=LI+1:GOSUB 1000:POK
E GH,PEEK(GH)-64:GOTO 980
940 IF CR=2 AND ZY<110 THEN POSITION 9
,4:? #6;"  ":GOSUB 1010:FOR QC=0 TO 4:
MM(QC)=0:F=1
950 IF F THEN F=0:POSITION AA(QC),BB(Q
C):? #6;" ":NEXT QC:POKE GH,PEEK(GH)-1
6
960 GOTO 980
970 POKE 1750,XA:POKE 1780,XB:GOSUB 43
0:POKE 53278,0:GOTO 370
980 POKE 53278,0:GOTO 370
990 SOUND 0,25,10,8:DE=1^1:SOUND 0,0,0
,0:POKE 656,1:POKE 657,28:? SC;:RETURN

1000 SOUND 0,200,10,8:DE=1^1:SOUND 0,0
,0,0:POKE 656,2:POKE 657,33:? LI;:RETU
RN
1010 SOUND 0,150,2,8:DE=1^1^1:SOUND 0,
0,0,0:SC=SC+500:POKE 656,1:POKE 657,28
:? SC;:RETURN
1020 XA=89:XB=145:LI=LI-1:SOUND 0,100,
2,8:SOUND 1,101,2,8:DE=1^1^1^1:SOUND 0
,0,0,0:SOUND 1,0,0,0
1030 CC=C:R=1:C=(CC<(SIZE+1))+(SIZE+1)
*(CC>SIZE AND CC<(SIZE*2+1))+(SIZE*2+1
)*(CC>(SIZE*2)):RETURN
1040 POSITION 15,6:? #6;CHR$(0):SOUND
0,100,10,8:DE=1^1:SOUND 0,0,0,0:POKE 6
56,2:POKE 657,33:? LI;:RETURN
1050 POKE 54286,64:POKE 53248,0:POKE 1
780,0:POKE 207,0:LI=LI-1:POKE 53278,0:
POKE 53249,0:RETURN
1060 GRAPHICS 1:POKE 53774,64:POKE 16,
64:TR=PEEK(710):SETCOLOR 2,0,0:DL=PEEK
(560)+256*PEEK(561)
1070 POKE DL+14,23:POKE DL+18,2:POKE 5
4276,4:SETCOLOR 0,8,4:POSITION 5,9:? #
6;"ROBOT DUNGEON"
1080 POSITION 4,11:? #6;"press start t
o begin":POSITION 9,13:? #6;"ANSCHUETZ
/WEISGERBER/ANSCHUETZ"
1090 IF PEEK(53279)=6 THEN POKE 710,TR
:GOTO 110
1100 GOTO 1090
1110 WX=INT((PEEK(1753)-45)/8)
1120 WY=INT((PEEK(1754)-27)/16)
1130 LOCATE WX,WY,WZ:FOR X=0 TO 4:IF W
Z=MM(X) THEN MM(X)=0:POSITION AA(X),BB
(X):? #6;CHR$(0):SC=SC+10:GOSUB 1150
1140 NEXT X:POKE 656,1:POKE 657,28:? S
C;:POKE 53278,0:POKE 1752,0:RETURN
1150 SOUND 0,200,12,10:DE=1^1:SOUND 0,
0,0,0:RETURN
1160 POKE 1756,0:POKE 53765,192:YX=PEE
K(1750):YY=PEEK(1780):POKE 559,0
1170 IF YX<64 THEN C=C-1:XA=184:XB=97:
GOSUB 430:GOTO 280
1180 IF YX>184 THEN C=C+1:XA=64:XB=97:
GOSUB 430:GOTO 280
1190 IF YX<128 THEN 1220
1200 IF YY<128 THEN C=C+(R/2<>INT(R/2)
):R=R+1:XA=89:XB=149:GOSUB 430:GOTO 28
0
1210 C=C+(R/2<>INT(R/2)):R=R-1:XA=89:X
B=47:GOSUB 430:GOTO 280
1220 IF YY<128 THEN C=C-(R/2=INT(R/2))
:R=R+1:XA=161:XB=149:GOSUB 430:GOTO 28
0
1230 C=C-(R/2=INT(R/2)):R=R-1:XA=161:X
B=47:GOSUB 430:GOTO 280
1240 POKE 54286,64:POKE 53765,192:POKE
 559,0
1250 COLOR 33:FOR SY=0 TO 9:PLOT 0,SY:
DRAWTO 19,SY:NEXT SY:COLOR 0:PLOT 0,0:
DRAWTO 5,0:DRAWTO 11,6
1260 DRAWTO 19,6:PLOT 0,1:DRAWTO 5,1:D
RAWTO 11,7:DRAWTO 19,7:PLOT 0,2:DRAWTO
 5,2:DRAWTO 10,7:PLOT 12,4
1270 DRAWTO 19,4:PLOT 12,5:DRAWTO 19,5
:COLOR 131:PLOT 11,6:DRAWTO 11,7
1280 POKE 1780,66:POKE 559,62:FOR PX=4
8 TO 88 STEP 8:POKE 53248,PX:GOSUB 147
0:FOR W=1 TO 75:NEXT W:NEXT PX
1290 FOR PY=82 TO 146 STEP 16:POKE 178
0,PY:POKE 53248,PX:GOSUB 1470:PX=PX+8:
FOR W=1 TO 75:NEXT W:NEXT PY
1300 FOR SX=31 TO 24 STEP -1:POKE CH*2
56+SX,0:FOR W=1 TO 15:NEXT W:SOUND 0,5
X+50,10,8:NEXT SX:SOUND 0,0,0,0
1310 POKE 53248,PX:GOSUB 1470:FOR W=1
TO 100:NEXT W:POKE 53248,PX+8:GOSUB 14
70
1320 FOR SX=24 TO 31:POKE CH*256+SX,56
:FOR W=1 TO 15:NEXT W:SOUND 0,SX+50,10
,8:NEXT SX:SOUND 0,0,0,0
1330 FOR PX=152 TO 200 STEP 8:POKE 532
48,PX:GOSUB 1470:FOR W=1 TO 75:NEXT W:
NEXT PX
1340 POKE 559,0:COLOR 0:PLOT 0,0:DRAWT
O 0,9:DRAWTO 19,9:DRAWTO 19,0:RETURN
1350 POKE 54286,64:POKE 53765,192:POKE
 559,0:MP=14
1360 COLOR 33:FOR SY=0 TO 9:PLOT 0,SY:
DRAWTO 19,SY:NEXT SY:COLOR 0:FOR SY=0
TO 7:PLOT 12,SY:DRAWTO 19,SY:NEXT SY
1370 FOR SY=4 TO 7:PLOT 0,SY:DRAWTO 7,
SY:NEXT SY:COLOR 131:FOR SY=6 TO 7:PLO
T 8,SY:DRAWTO 11,SY:NEXT SY
1380 POSITION INT((20-LEN(A$))/2),9:?
#6;A$
1390 POKE 1780,146:POKE 559,62:FOR PX=
48 TO 104 STEP 8:POKE 53248,PX:GOSUB 1
470:FOR W=1 TO 75:NEXT W:NEXT PX
1400 FOR SX=31 TO 24 STEP -1:POKE CH*2
56+SX,0:FOR W=1 TO 15:NEXT W:SOUND 0,5
X+50,10,8:NEXT SX:SOUND 0,0,0,0
1410 FOR PX=112 TO 144 STEP 8:POKE 532
48,PX:GOSUB 1470:FOR W=1 TO 75:NEXT W:
NEXT PX
1420 FOR SX=24 TO 31:POKE CH*256+SX,56
:FOR W=1 TO 15:NEXT W:SOUND 0,SX+50,10
,8:NEXT SX:SOUND 0,0,0,0
1430 POKE 53248,152:GOSUB 1470
1440 POSITION 14,1:? #6;"YOU":POSITION
 14,2:? #6;"MADE":POSITION 14,3:? #6;"
IT<<<"
1450 FOR EX=1 TO 4:POKE 1780,139:FOR W
=1 TO 10:NEXT W:POKE 1780,146:GOSUB 14
70:FOR W=1 TO 10:NEXT W:NEXT EX
1460 GOTO 1460
1470 SOUND 0,66,12,8:FOR WN=1 TO 15:NE
XT WN:SOUND 0,0,0,0
1480 MP=MP-1:IF MP<9 AND MP>0 THEN POS
ITION 0,7:? #6;M$(MP,8)
1490 RETURN
1500 POKE 54286,64:POKE 53765,192:? CH
R$(125)
1510 IF D$(1,1)="D" THEN 1530
1520 ? "PRESS RECORD AND PLAY ON RECOR
DER     AND THEN PRESS RETURN ON KEYBO
ARD":GOTO 1540
1530 ? "READY DISK AND THEN PRESS RETU
RN      ON KEYBOARD":POKE 764,255:GOSU
B 1600
1540 POKE 764,255:OPEN #1,8,0,D$:PRINT
 #1,SIZE
1545 FOR NR=1 TO SIZE:FOR NC=1 TO SIZE
*3:NA=PEEK(FP+(NR-1)*SIZE*3+NC):PUT #1
,NA:NEXT NC:NEXT NR
1550 FOR NR=1 TO SIZE:FOR NC=1 TO SIZE
*3:NA=PEEK(SP+(NR-1)*SIZE*3+NC):PUT #1
,NA:NEXT NC:NEXT NR
1560 FOR P=8 TO 127:NA=PEEK(CH*256+P):
PUT #1,NA:NEXT P
1570 PUT #1,LEN(A$):FOR P=1 TO LEN(A$)
:PUT #1,ASC(A$(P,P)):NEXT P:PUT #1,0
1580 PRINT #1,SC:PUT #1,R:PUT #1,C:PRI
NT #1,LI:PUT #1,XA:PUT #1,XB:CLOSE #1:
GOSUB 1730:POKE 54286,192
1590 RETURN
1600 IF PEEK(764)<255 THEN RETURN
1610 GOTO 1600
1620 PM=(MEM-12)*256:FOR X=PM TO PM+13
:READ A:POKE X,A:NEXT X:FOR X=PM+256 T
O PM+261:READ A:POKE X,A:NEXT X
1630 FOR X=1536 TO 1706:READ A:POKE X,
A:NEXT X:FOR X=(MEM-10)*256 TO (MEM-10
)*256+248:READ A:POKE X,A
1640 IF A=86 THEN POKE X,MEM-10
1650 NEXT X:FOR X=(MEM-2)*256 TO (MEM-
2)*256+68:READ A:POKE X,A
1660 IF A=86 THEN POKE X,MEM-10
1670 NEXT X:POKE (MEM-10)*256+243,MEM-
2
1680 FOR X=207 TO 209:POKE X,0:NEXT X:
POKE 623,1:POKE 1788,PM/256:POKE 53277
,3:POKE 54279,PM/256-4
1690 POKE 1784,14:POKE 1785,6:POKE 175
0,89:POKE 1780,145:POKE 704,52:POKE 70
```

```
5,194:POKE 1752,0:X=USR(1696)
1700 FOR X=PM TO PM+13:POKE X,0:NEXT X

1710 DL=PEEK(560)+256*PEEK(561):POKE D
L+15,194:POKE DL+18,130:POKE DL+19,130
:POKE DL+20,130
1720 POKE 512,111:POKE 513,MEM-10:POKE
 752,1:POKE 1756,1:IF STAT THEN XA=89:
XB=145:LI=10:R=1:C=1:SC=0
1730 ? CHR$(125):TJ=PEEK(660)+256*PEEK
(661):FOR EA=0 TO 39:POKE EA+TJ,1:POKE
 EA+TJ+120,1:NEXT EA
1740 FOR EA=40 TO 80 STEP 40:POKE TJ+E
A,1:POKE TJ+EA+39,1:NEXT EA
1750 POKE 656,1:POKE 657,2:? A$;:POKE
656,1:POKE 657,21:? "SCORE: ";SC;
1760 POKE 656,2:POKE 657,2:? "ROOM NUM
BER ";RN;" ";:POKE 656,2:POKE 657,21:?
 "HIT POINTS: ";LI;:RETURN
1770 DATA 0,0,0,224,160,191,170,234,3,
3,15,15,63,63,255,255,0,24,24,24,24,0,
24,0
1780 DATA 0,56,254,84,40,56,146,124,56
,56,56,40,40,108,0,0,0,0,0,16
1790 DATA 162,3,189,244,6,240,89,56,22
1,240,6,240,83,141,254,6,106,141,255,6
,142,253,6,24,169,0,109,253,6,24,109
1800 DATA 252,6,133,204,133,206,189,24
0,6,133,203,173,254,6,133,205,189,248,
6,170,232,46,255,6,144,16,168,177
1810 DATA 203,145,205,169,0,145,203,13
6,202,208,244,76,87,6,160,0,177,203,14
5,205,169,0,145,203,200,202,208,244
1820 DATA 174,253,6,173,254,6,157,240,
6,189,236,6,240,48,133,203,24,138,141,
253,6,109,235,6,133,204,24,173,253
1830 DATA 6,109,252,6,133,206,189,240,
6,133,205,189,248,6,170,160,0,177,203,
145,205,200,202,208,248,174,253,6,169
1840 DATA 0,157,236,6,202,48,3,76,2,6,
76,98,228,0,0,104,169,7,162,6,160,0,32
,92,228,96
1850 DATA 72,138,72,165,207,201,1,240,
34,173,132,2,201,1,240,51,173,120,2,20
1,15,240,44,173,214,6,133,209,173,244
1860 DATA 6,141,245,6,173,120,2,133,20
8,169,1,133,207,216,24,166,208,165,209
,125,79,86,133,209,141,1,208,24,173
1870 DATA 245,6,125,95,86,141,245,6,14
1,10,212,169,111,141,0,2,104,170,104,6
4,0,0,0,0,0,1,1,1,0,255,255,255
1880 DATA 0,0,0,0,0,0,0,0,0,0,1,255,0,0,
1,255,0,0,1,255,0
1890 DATA 72,138,72,173,215,6,201,1,24
0,35,169,1,141,215,6,216,24,174,120,2,
189,79,86,109,214,6,141,214,6
1900 DATA 141,0,208,24,189,95,86,109,2
44,6,141,244,6,24,144,5,169,0,141,215,
6,141,10,212,169,173,141,0,2
1910 DATA 104,170,104,64,72,173,5,208,
201,0,240,48,201,4,240,15,169,0,234,23
4,141,1,208,133,207,234,234,234
1920 DATA 24,144,29,201,4,208,25,165,2
09,141,217,6,173,245,6,141,218,6,169,0
,234,234,141,1,208,133,207,169,1
1930 DATA 141,216,6,169,0,234,234,234,
141,10,212,169,0,141,0,2,169,94,141,1,
2,104,64
1940 DATA 72,173,220,6,201,1,240,46,21
6,173,219,6,24,105,1,141,219,6,201,60,
208,13,169,82,141,4,210,169,193,141,5
1950 DATA 210,24,144,19,201,120,208,15
,169,90,141,4,210,169,193,141,5,210,16
9,0,141,219,6,141,10,212,169,0,141,0,2

1960 DATA 169,86,141,1,2,104,64
```

# ROBOT DUNGEON – DUNGEON CREATOR

```
10 REM DUNGEON CREATOR
20 REM BY ANSCHUETZ/WEISGERBER/ANSCHUE
TZ
30 REM (c) 1985, ANTIC PUBLISHING
40 ? "WARNING! The DUNGEON CREATOR"

50 ? "WRITES TO YOUR DISK!!!"
60 ? :? "Do you want instructions (Y/N
)?";
62 POKE 764,255:POKE 1540,240
65 IF (PEEK(764)=43 OR PEEK(764)=107)
THEN POKE 764,255:RUN "D:ROBOHELP"
67 IF (PEEK(764)=35 OR PEEK(764)=99) T
HEN POKE 764,255:GOTO 100
70 GOTO 65
100 DIM D$(20,60),R$(20,60),M(10),A$(1
5):GRAPHICS 2:POKE 16,64:POKE 53774,64
:POSITION 2,4:? #6;"DUNGEON CREATOR"
110 POSITION 7,6:? #6;"A/W/A":DIM D$(1
2)
120 ? "ENTER THE NAME OF YOUR DUNGEON,
      EXCLUDING ALL PUCTUATION, AND L
IMITED TO 15 CHARACTERS";:INPUT A$
130 ? CHR$(125):? "ENTER FILENAME.  CA
SSETTE OWNERS USE  SIMPLY C:  DISK USE
RS USE D:DUNJIN.DAT"
140 TRAP 130:INPUT D$:IF D$(1,1)<>"C"
AND D$(1,1)<>"D" THEN 130
150 FOR R=1 TO 20:FOR C=1 TO 60:D$(R,C
)=0:R$(R,C)=0:NEXT C:NEXT R
160 FOR L=1 TO 3:HI=L*20:LOW=HI-19:R=1
:C=INT(RND(0)*20)+LOW
170 FOR JW=1 TO 399
180 POKE 77,0:? L,JW,R,C:Q=0:RA=(R/2=I
NT(R/2))
190 IF C>LOW THEN IF D$(R,C-1)=0 THEN
Q=Q+1:M(Q)=1
200 IF C<HI THEN IF D$(R,C+1)=0 THEN Q
=Q+1:M(Q)=2
210 IF (C>LOW OR RA=0) AND R<20 THEN I
F D$(R+1,C-RA)=0 THEN Q=Q+1:M(Q)=3
220 IF (C<HI OR RA=1) AND R<20 THEN IF
 D$(R+1,C+(RA=0))=0 THEN Q=Q+1:M(Q)=4
230 IF (C>LOW OR RA=0) AND R>1 THEN IF
 D$(R-1,C-RA)=0 THEN Q=Q+1:M(Q)=5
240 IF (C>HI OR RA=1) AND R>1 THEN IF
D$(R-1,C+(RA=0))=0 THEN Q=Q+1:M(Q)=6
250 IF Q>0 THEN 290
260 C=C+1:IF C>HI THEN C=LOW:R=R+1:IF
R>20 THEN R=1
270 IF D$(R,C)=0 THEN 260
280 GOTO 180
290 EA=M(INT(RND(0)*Q)+1)
300 IF EA=1 THEN D$(R,C)=D$(R,C)+1:C=C
-1:D$(R,C)=D$(R,C)+16:GOTO 360
310 IF EA=2 THEN D$(R,C)=D$(R,C)+16:C=
C+1:D$(R,C)=D$(R,C)+1:GOTO 360
320 IF EA=3 THEN D$(R,C)=D$(R,C)+32:R=
R+1:C=C-RA:D$(R,C)=D$(R,C)+4:GOTO 360
330 IF EA=4 THEN D$(R,C)=D$(R,C)+64:R=
R+1:C=C+(RA=0):D$(R,C)=D$(R,C)+2:GOTO
360
340 IF EA=5 THEN D$(R,C)=D$(R,C)+2:R=R
-1:C=C-RA:D$(R,C)=D$(R,C)+64:GOTO 360
350 IF EA=6 THEN D$(R,C)=D$(R,C)+4:R=R
-1:C=C+(RA=0):D$(R,C)=D$(R,C)+32:GOTO
360
360 NEXT JW:NEXT L:? CHR$(125):? A$;"
IS ABOUT HALF-WAY":? "COMPLETED."
370 FOR L=1 TO 3:FOR X=1 TO 25:GOSUB 7
80
380 IF D$(R,C)-128>0 THEN X=X-1:GOTO 4
00
390 D$(R,C)=D$(R,C)+128
400 NEXT X:NEXT L
410 FOR L=1 TO 3:FOR X=1 TO 200:GOSUB
780:IF R$(R,C)>0 THEN X=X-1:GOTO 430
420 R$(R,C)=R$(R,C)+1
430 NEXT X:NEXT L
440 FOR L=1 TO 3:FOR X=1 TO 200:GOSUB
780:IF R$(R,C)>1 THEN X=X-1:GOTO 460
450 R$(R,C)=R$(R,C)+2
460 NEXT X:NEXT L
470 FOR L=1 TO 3:FOR X=1 TO 100:GOSUB
780:IF R$(R,C)>3 THEN X=X-1:GOTO 490
480 R$(R,C)=R$(R,C)+4
490 NEXT X:NEXT L
500 FOR L=1 TO 3:FOR X=1 TO 80:GOSUB 7
80:IF R$(R,C)>7 THEN X=X-1:GOTO 520
510 R$(R,C)=R$(R,C)+8
520 NEXT X:NEXT L
530 FOR L=1 TO 3:FOR X=1 TO 25:GOSUB 7
80:IF R$(R,C)>15 THEN X=X-1:GOTO 550
540 R$(R,C)=R$(R,C)+16
550 NEXT X:NEXT L
560 FOR L=1 TO 3:FOR X=1 TO 25:GOSUB 7
80:IF R$(R,C)>31 THEN X=X-1:GOTO 580
570 R$(R,C)=R$(R,C)+32
580 NEXT X:NEXT L
590 FOR R=1 TO 20:FOR C=1 TO 60:R$(R,C
)=R$(R,C)+64:NEXT C:NEXT R
600 FOR L=1 TO 3:FOR X=1 TO 200:GOSUB
780:IF R$(R,C)>127 THEN X=X-1:GOTO 620

610 R$(R,C)=R$(R,C)+128
620 NEXT X:NEXT L:POKE 764,255
630 POKE 764,255
640 IF D$(1,1)="C" THEN ? "PRESS RECOR
D/PLAY ON RECORDER AND THENPRESS RETUR
N ON KEYBOARD":OPEN #1,8,0,D$
650 IF D$(1,1)="D" THEN ? CHR$(125):?
"READY DISK AND THEN PRESS RETURN ON
 KEYBOARD":GOSUB 670:OPEN #1,8,0,D$
660 GOTO 690
670 IF PEEK(764)<255 THEN RETURN
680 GOTO 670
690 FOR R=1 TO 20:FOR C=1 TO 60:A=D$(R
,C):PUT #1,A:NEXT C:NEXT R
700 FOR R=1 TO 20:FOR C=1 TO 60:A=R$(R
,C):PUT #1,A:NEXT C:NEXT R
710 FOR X=1 TO 120:READ A:PUT #1,A:NEX
T X:PUT #1,LEN(A$):FOR X=1 TO LEN(A$):
PUT #1,ASC(A$(X,X)):NEXT X
720 PUT #1,1:PRINT #1,0:PUT #1,0:PUT #
1,0:PRINT #1,0:PUT #1,0:PUT #1,0:CLOSE
 #1:END
730 DATA 221,221,170,119,119,119,170,2
21,0,0,0,255,255,255,0,0,56,56,56,56,5
6,56,56,56,16,168,18,72,16,0,0,0
740 DATA 0,0,0,0,14,31,14,4,0,0,0,170,
85,0,0,0,0,0,108,146,146,108,0,0
750 DATA 3,4,9,11,11,9,4,3,252,2,249,1
3,13,249,2,252
760 DATA 126,153,153,126,24,60,66,165,
24,36,255,189,60,60,60,231,60,66,219,6
6,60,231,129,231
770 DATA 36,24,189,165,126,60,60,231,2
4,60,102,60,24,126,219,153,0,0,254,254
,65,127,119,127
780 R=INT(RND(0)*20)+1:C=INT(RND(0)*20
)+L*20-19:RETURN
```

# ROBOT DUNGEON – DUNGEON CREATOR - 2017

```
5 REM MODIFIED ON 7/31/2017 BY ERIC AN
SCHUETZ TO PRODUCE VARIABLE SIZED DUNG
EONS FROM 5 TO 20
7 REM MODIFIED CODE HAS VARIABLES MAXS
IZE, SIZE, AND BASENUM
10 REM DUNGEON CREATOR
20 REM BY ANSCHUETZ/WEISGERBER/ANSCHUE
TZ
30 REM (c) 1985, ANTIC PUBLISHING
40 ? "{RVS}WARNING! The {RVS} DUNGEON CREATOR {RVS} "
50 ? "WRITES TO YOUR DISK!!!"
60 ? :? "Do you want instructions (Y/N
)?";
62 POKE 764,255:POKE 1540,240
65 IF (PEEK(764)=43 OR PEEK(764)=107)
THEN POKE 764,255:RUN "D:ROBOHELP"
67 IF (PEEK(764)=35 OR PEEK(764)=99) T
HEN POKE 764,255:GOTO 90
70 GOTO 65
90 MAXSIZE=20
100 DIM DS(MAXSIZE,MAXSIZE*3),RS(MAXSI
ZE,MAXSIZE*3)
105 DIM M(10),A$(15):GRAPHICS 2:POKE 1
6,64:POKE 53774,64:POSITION 2,4:? #6;"
{RVS} DUNGEON CREATOR {RVS}"
110 POSITION 7,6:? #6;"A/W/A":DIM D$(1
2)
120 ? "ENTER THE NAME OF YOUR DUNGEON,
      EXCLUDING ALL PUNCTUATION, AND
LIMITED TO 15 CHARACTERS";:INPUT A$
125 ? CHR$(125):? "ENTER THE SIZE OF T
HE DUNGEON (5-20)"
127 TRAP 125:INPUT SIZE:IF SIZE<5 OR S
IZE>20 THEN 125
130 ? CHR$(125):? "ENTER FILENAME.  CA
SSETTE OWNERS USE  SIMPLY C:  DISK USE
RS USE D:DUNJIN.DAT"
140 TRAP 130:INPUT D$:IF D$(1,1)<>"C"
AND D$(1,1)<>"D" THEN 130
150 FOR R=1 TO SIZE:FOR C=1 TO SIZE*3:
DS(R,C)=0:RS(R,C)=0:NEXT C:NEXT R
160 FOR L=1 TO 3:HI=L*SIZE:LOW=HI-(SIZ
E-1):R=1:C=INT(RND(0)*SIZE)+LOW
170 FOR JW=1 TO SIZE*SIZE-1
180 POKE 77,0:? L,JW,R,C:Q=0:RA=(R/2=I
NT(R/2))
190 IF C>LOW THEN IF DS(R,C-1)=0 THEN
Q=Q+1:M(Q)=1
200 IF C<HI THEN IF DS(R,C+1)=0 THEN Q
=Q+1:M(Q)=2
210 IF (C>LOW OR RA=0) AND R<SIZE THEN
 IF DS(R+1,C-RA)=0 THEN Q=Q+1:M(Q)=3
220 IF (C<HI OR RA=1) AND R<SIZE THEN
IF DS(R+1,C+(RA=0))=0 THEN Q=Q+1:M(Q)=
4
230 IF (C>LOW OR RA=0) AND R>1 THEN IF
 DS(R-1,C-RA)=0 THEN Q=Q+1:M(Q)=5
240 IF (C>HI OR RA=1) AND R>1 THEN IF
DS(R-1,C+(RA=0))=0 THEN Q=Q+1:M(Q)=6
250 IF Q>0 THEN 290
260 C=C+1:IF C>HI THEN C=LOW:R=R+1:IF
R>SIZE THEN R=1
270 IF DS(R,C)=0 THEN 260
280 GOTO 180
290 EA=M(INT(RND(0)*Q)+1)
300 IF EA=1 THEN DS(R,C)=DS(R,C)+1:C=C
-1:DS(R,C)=DS(R,C)+16:GOTO 360
310 IF EA=2 THEN DS(R,C)=DS(R,C)+16:C=
C+1:DS(R,C)=DS(R,C)+1:GOTO 360
320 IF EA=3 THEN DS(R,C)=DS(R,C)+32:R=
R+1:C=C-RA:DS(R,C)=DS(R,C)+4:GOTO 360
330 IF EA=4 THEN DS(R,C)=DS(R,C)+64:R=
R+1:C=C+(RA=0):DS(R,C)=DS(R,C)+2:GOTO
360
340 IF EA=5 THEN DS(R,C)=DS(R,C)+2:R=R
-1:C=C-RA:DS(R,C)=DS(R,C)+64:GOTO 360
350 IF EA=6 THEN DS(R,C)=DS(R,C)+4:R=R
-1:C=C+(RA=0):DS(R,C)=DS(R,C)+32:GOTO
360
360 NEXT JW:NEXT L:? CHR$(125):? A$;"
IS ABOUT HALF-WAY":? "COMPLETED."
365 BASENUM=INT(SIZE*SIZE/16):REM EEA
THE FOLLOWING CODE USES BASENUM TO SCA
LE NUMBER OF TREASURES BASED ON SIZE
370 ? "1 ";:FOR L=1 TO 3:FOR X=1 TO BA
SENUM:GOSUB 780
380 IF DS(R,C)-128>0 THEN X=X-1:GOTO 4
00
390 DS(R,C)=DS(R,C)+128
400 NEXT X:NEXT L
410 ? "2 ";:FOR L=1 TO 3:FOR X=1 TO BA
SENUM*8:GOSUB 780:IF RS(R,C)>0 THEN X=
X-1:GOTO 430
420 RS(R,C)=RS(R,C)+1
430 NEXT X:NEXT L
440 ? "3 ";:FOR L=1 TO 3:FOR X=1 TO BA
SENUM*8:GOSUB 780:IF RS(R,C)>1 THEN X=
X-1:GOTO 460
450 RS(R,C)=RS(R,C)+2
460 NEXT X:NEXT L
470 ? "4 ";:FOR L=1 TO 3:FOR X=1 TO BA
SENUM*3:GOSUB 780:IF RS(R,C)>3 THEN X=
X-1:GOTO 490
480 RS(R,C)=RS(R,C)+4
490 NEXT X:NEXT L
500 ? "5 ";:FOR L=1 TO 3:FOR X=1 TO BA
SENUM*3:GOSUB 780:IF RS(R,C)>7 THEN X=
X-1:GOTO 520
510 RS(R,C)=RS(R,C)+8
520 NEXT X:NEXT L
530 ? "6 ";:FOR L=1 TO 3:FOR X=1 TO BA
SENUM:GOSUB 780:IF RS(R,C)>15 THEN X=X
-1:GOTO 550
540 RS(R,C)=RS(R,C)+16
550 NEXT X:NEXT L
560 ? "7 ";:FOR L=1 TO 3:FOR X=1 TO BA
SENUM:GOSUB 780:IF RS(R,C)>31 THEN X=X
-1:GOTO 580
570 RS(R,C)=RS(R,C)+32
580 NEXT X:NEXT L
590 FOR R=1 TO SIZE:FOR C=1 TO SIZE*3:
RS(R,C)=RS(R,C)+64:NEXT C:NEXT R
600 ? "8 ":FOR L=1 TO 3:FOR X=1 TO BAS
ENUM*8:GOSUB 780:IF RS(R,C)>127 THEN X
=X-1:GOTO 620
610 RS(R,C)=RS(R,C)+128
620 NEXT X:NEXT L:POKE 764,255
630 POKE 764,255
640 IF D$(1,1)="C" THEN ? "PRESS {RVS}RECOR
D/PLAY{RVS} ON RECORDER AND THENPRESS {RVS}RETUR
N{RVS} ON KEYBOARD":OPEN #1,8,0,D$
650 IF D$(1,1)="D" THEN ? CHR$(125):?
"{RVS}READY{RVS} DISK AND THEN PRESS {RVS}RETURN{RVS} ON
 KEYBOARD":GOSUB 670:OPEN #1,8,0,D$
660 GOTO 685
670 IF PEEK(764)<255 THEN RETURN
680 GOTO 670
685 PRINT #1,SIZE
690 FOR R=1 TO SIZE:FOR C=1 TO SIZE*3:
A=DS(R,C):PUT #1,A:NEXT C:NEXT R
700 FOR R=1 TO SIZE:FOR C=1 TO SIZE*3:
A=RS(R,C):PUT #1,A:NEXT C:NEXT R
710 FOR X=1 TO 120:READ A:PUT #1,A:NEX
T X:PUT #1,LEN(A$):FOR X=1 TO LEN(A$):
PUT #1,ASC(A$(X,X)):NEXT X
720 PUT #1,1:PRINT #1,0:PUT #1,0:PUT #
1,0:PRINT #1,0:PUT #1,0:PUT #1,0:CLOSE
 #1:END
730 DATA 221,221,170,119,119,119,170,2
21,0,0,0,255,255,255,0,0,56,56,56,56,5
6,56,56,56,16,168,18,72,16,0,0,0
740 DATA 0,0,0,0,14,31,14,4,0,0,0,170,
85,0,0,0,0,0,0,108,146,146,108,0,0
750 DATA 3,4,9,11,11,9,4,3,252,2,249,1
3,13,249,2,252
760 DATA 126,153,153,126,24,60,66,165,
24,36,255,189,60,60,60,231,60,66,219,6
6,60,231,129,231
770 DATA 36,24,189,165,126,60,60,231,2
4,60,102,60,24,126,219,153,0,0,254,254
,65,127,119,127
780 R=INT(RND(0)*SIZE)+1:C=INT(RND(0)*
SIZE)+L*SIZE-(SIZE-1):RETURN
```

# ROBOT DUNGEON – ROBOHELP

```
10 DIM LINE$(120),A$(1)
20 ? "ß":CLOSE #1:OPEN #1,4,0,"D:ROBO.
TXT"
30 TRAP 100
40 INPUT #1,LINE$
50 IF LEN(LINE$)>2 THEN IF LINE$(1,3)=
"XXX" THEN ? :? :POSITION 2,22:? "PRES
S RETURN ";:INPUT A$:? "ß":GOTO 40
60 ? LINE$:GOTO 40
100 CLOSE #1:TRAP 40000:POSITION 2,22:
? "RETURN TO RUN PROGRAM ";:INPUT A$
110 IF PEEK(1540)=240 THEN RUN "D:DUNG
EON.CRT"
120 RUN "D:ROBOT5.BAS"
```

# SOKO-BAN

```
1 MAXLEV=55
3 GOSUB 10500
4 DIM CH$(1),SCREEN$(320),SAVSCRN$(320
),FILNAM$(14),TEN$(1),WON$(1),LINE$(20
),BEGSCR$(320)
5 DIM RM$(3),RP$(3),RT$(3),NAM$(3)
6 GOSUB 900
7 TRAP 7:GRAPHICS 1:POKE 16,64:POKE 53
774,64:POKE 752,1:POKE 756,224:? :? "
     ENTER NAME (3 INITIALS)";
8 INPUT NAM$
10 TRAP 10:GRAPHICS 1:POKE 16,64:POKE
53774,64:POKE 752,1:POKE 756,224
11 ? :? "                ENTER LEVEL";:INP
UT LEVEL
12 IF LEVEL<1 OR LEVEL>MAXLEV THEN ? "
ENTER 1-";MAXLEV;". PRESS RETURN TO CO
NTINUE";:TRAP 10:INPUT X:GOTO 10
13 GOSUB 10800:IF STAT=2 THEN GOTO 10
16 POKE 18,0:POKE 19,0:POKE 20,0
17 PUSHES=0:MOVES=0:PLACED=0
18 BEGSCR$=SCREEN$:SX=XPOS:SY=YPOS:SB=
BOXES:SAVSCRN$=SCREEN$:PITCH=30:OLDX=X
POS:OLDY=YPOS:PUSH=-1
19 POKE 764,255:GOSUB 10740:? :? :? :?
 :POKE 756,CHSET/256
20 SETCOLOR 0,3,2:SETCOLOR 1,13,4:SETC
OLOR 2,8,2:SETCOLOR 3,2,6:SETCOLOR 4,0
,0
30 FOR I=1 TO 16
40 POSITION 0,I+1
50 ? #6;SCREEN$((I-1)*20+1,(I*20));
60 NEXT I
80 ? "                    ";LEVEL;" ";:?
:?
100 POKE 77,0:POSITION XPOS-1,YPOS+1:?
 #6;"a";:IF PEEK(764)=10 THEN GOSUB 10
900:GOTO 19
101 IF PEEK(764)=28 THEN GOTO 7
102 GOSUB 10600:POSITION 6,19:? #6;"0"
;HOURS;":";:IF MINS<10 THEN ? #6;"0";M
INS;":";:GOTO 104
103 ? #6;MINS;":";
104 IF SECS<10 THEN ? #6;"0";SECS;:GOT
O 106
105 ? #6;SECS;
106 IF PLACED=BOXES THEN GOSUB 1080:OL
DLEV=LEVEL:LEVEL=LEVEL+1:GOTO 13
107 IF PEEK(53279)=6 THEN SCREEN$=BEGS
CR$:XPOS=SX:YPOS=SY:BOXES=SB:GOTO 17
108 ST=STICK(0):IF ST=11 OR ST=7 OR ST
=14 OR ST=13 THEN SAVSCRN$=SCREEN$:OLD
X=XPOS:OLDY=YPOS
109 SCRPOS=(YPOS-1)*20+XPOS:CH$=SCREEN
$(SCRPOS,SCRPOS)
110 IF ST=11 THEN GOSUB 500
120 IF ST=7 THEN GOSUB 600
130 IF ST=14 THEN GOSUB 700
140 IF ST=13 THEN GOSUB 800
150 IF STRIG(0)=1 THEN GOTO 100
155 SCREEN$=SAVSCRN$:XPOS=OLDX:YPOS=OL
DY
160 FOR I=1 TO 16:POSITION 0,I+1:? #6;
SCREEN$((I-1)*20+1,(I*20));:NEXT I
161 IF PUSH=-1 THEN GOTO 170
162 IF PUSH=1 THEN PUSHES=PUSHES-1
164 MOVES=MOVES-1:GOSUB 10740
165 PUSH=-1
170 GOTO 100
500 IF SCREEN$(SCRPOS-1,SCRPOS-1)="B"
THEN RETURN
505 IF SCREEN$(SCRPOS-1,SCRPOS-1)=" "
THEN GOTO 515
510 IF SCREEN$(SCRPOS-1,SCRPOS-1)="D"
THEN SCREEN$(SCRPOS-1,SCRPOS-1)="V":GO
TO 515
512 GOTO 520
515 IF SCREEN$(SCRPOS,SCRPOS)="a" THEN
 SCREEN$(SCRPOS,SCRPOS)=" "
517 IF SCREEN$(SCRPOS,SCRPOS)="V" THEN
 SCREEN$(SCRPOS,SCRPOS)="D"
518 POSITION XPOS-1,YPOS+1:? #6;"e";:F
OR I=1 TO 5:NEXT I:POSITION XPOS-1,YPO
S+1:? #6;SCREEN$(SCRPOS,SCRPOS);
519 POSITION XPOS-1-1,YPOS+1:? #6;"e";
:POSITION XPOS-1-1,YPOS+1:? #6;"a";:XP
OS=XPOS-1:GOSUB 1010:PUSH=0:RETURN
520 IF SCREEN$(SCRPOS-2,SCRPOS-2)="B"
THEN RETURN
525 IF SCREEN$(SCRPOS-2,SCRPOS-2)="c"
THEN RETURN
527 IF SCREEN$(SCRPOS-2,SCRPOS-2)="z"
THEN RETURN
530 IF SCREEN$(SCRPOS-2,SCRPOS-2)="D"
THEN SCREEN$(SCRPOS-2,SCRPOS-2)="z":PL
ACED=PLACED+1
535 IF SCREEN$(SCRPOS-2,SCRPOS-2)=" "
THEN SCREEN$(SCRPOS-2,SCRPOS-2)="c"
550 IF SCREEN$(SCRPOS-1,SCRPOS-1)="z"
THEN SCREEN$(SCRPOS-1,SCRPOS-1)="V":PL
ACED=PLACED-1:GOTO 560
555 SCREEN$(SCRPOS-1,SCRPOS-1)="a"
560 IF SCREEN$(SCRPOS,SCRPOS)="V" THEN
 SCREEN$(SCRPOS,SCRPOS)="D"
570 IF SCREEN$(SCRPOS,SCRPOS)="a" THEN
 SCREEN$(SCRPOS,SCRPOS)=" "
575 POSITION XPOS-1,YPOS+1:? #6;"e";:G
OSUB 1000
577 POSITION XPOS-1-2,YPOS+1:? #6;"fg"
;
579 IF SCREEN$(SCRPOS,SCRPOS)="c" THEN
 POSITION XPOS-1,YPOS+1:? #6;"h";:GOTO
 582
580 POSITION XPOS-1,YPOS+1:? #6;"i";
582 GOSUB 1000:POSITION XPOS-2-1,YPOS+
1:? #6;"ce";SCREEN$(SCRPOS,SCRPOS);:GO
SUB 1000
584 POSITION XPOS-1-1,YPOS+1:? #6;"a";

585 PUSH=1:XPOS=XPOS-1:GOSUB 10700:RET
URN
600 IF SCREEN$(SCRPOS+1,SCRPOS+1)="B"
THEN RETURN
605 IF SCREEN$(SCRPOS+1,SCRPOS+1)=" "
THEN GOTO 615
610 IF SCREEN$(SCRPOS+1,SCRPOS+1)="D"
THEN SCREEN$(SCRPOS+1,SCRPOS+1)="V":GO
TO 615
612 GOTO 620
615 IF SCREEN$(SCRPOS,SCRPOS)="a" THEN
 SCREEN$(SCRPOS,SCRPOS)=" "
617 IF SCREEN$(SCRPOS,SCRPOS)="V" THEN
 SCREEN$(SCRPOS,SCRPOS)="D"
618 POSITION XPOS-1,YPOS+1:? #6;"f";:F
OR I=1 TO 5:NEXT I:POSITION XPOS-1,YPO
S+1:? #6;SCREEN$(SCRPOS,SCRPOS);
619 POSITION XPOS+1-1,YPOS+1:? #6;"f";
:POSITION XPOS+1-1,YPOS+1:? #6;"a";:XP
OS=XPOS+1:GOSUB 1010:PUSH=0:RETURN
620 IF SCREEN$(SCRPOS+2,SCRPOS+2)="B"
THEN RETURN
625 IF SCREEN$(SCRPOS+2,SCRPOS+2)="c"
THEN RETURN
627 IF SCREEN$(SCRPOS+2,SCRPOS+2)="z"
THEN RETURN
630 IF SCREEN$(SCRPOS+2,SCRPOS+2)="D"
THEN SCREEN$(SCRPOS+2,SCRPOS+2)="z":PL
ACED=PLACED+1
635 IF SCREEN$(SCRPOS+2,SCRPOS+2)=" "
THEN SCREEN$(SCRPOS+2,SCRPOS+2)="c"
650 IF SCREEN$(SCRPOS+1,SCRPOS+1)="z"
THEN SCREEN$(SCRPOS+1,SCRPOS+1)="V":PL
ACED=PLACED-1:GOTO 660
655 SCREEN$(SCRPOS+1,SCRPOS+1)="a"
660 IF SCREEN$(SCRPOS,SCRPOS)="V" THEN
 SCREEN$(SCRPOS,SCRPOS)="D"
670 IF SCREEN$(SCRPOS,SCRPOS)="a" THEN
 SCREEN$(SCRPOS,SCRPOS)=" "
675 POSITION XPOS-1,YPOS+1:? #6;"f";:G
OSUB 1000
677 IF SCREEN$(SCRPOS,SCRPOS)="c" THEN
 POSITION XPOS-1,YPOS+1:? #6;"l";:GOTO
 681
680 POSITION XPOS-1,YPOS+1:? #6;"k";
681 POSITION XPOS+1-1,YPOS+1:? #6;"mn"
;
682 GOSUB 1000:POSITION XPOS-1,YPOS+1:
? #6;SCREEN$(SCRPOS,SCRPOS);"fc";:GOSU
B 1000
684 POSITION XPOS+1-1,YPOS+1:? #6;"a";

685 PUSH=1:XPOS=XPOS+1:GOSUB 10700:RET
URN
700 IF SCREEN$(SCRPOS-20,SCRPOS-20)="B
" THEN RETURN
705 IF SCREEN$(SCRPOS-20,SCRPOS-20)="
" THEN GOTO 715
710 IF SCREEN$(SCRPOS-20,SCRPOS-20)="D
" THEN SCREEN$(SCRPOS-20,SCRPOS-20)="V
":GOTO 715
712 GOTO 720
715 IF SCREEN$(SCRPOS,SCRPOS)="a" THEN
 SCREEN$(SCRPOS,SCRPOS)=" "
717 IF SCREEN$(SCRPOS,SCRPOS)="V" THEN
 SCREEN$(SCRPOS,SCRPOS)="D"
718 POSITION XPOS-1,YPOS+1:? #6;"d";:F
OR I=1 TO 5:NEXT I:POSITION XPOS-1,YPO
S+1:? #6;SCREEN$(SCRPOS,SCRPOS);
719 POSITION XPOS-1,YPOS-1+1:? #6;"d";
:POSITION XPOS-1,YPOS-1+1:? #6;"a";:YP
OS=YPOS-1:GOSUB 1010:PUSH=0:RETURN
720 IF SCREEN$(SCRPOS-40,SCRPOS-40)="B
" THEN RETURN
725 IF SCREEN$(SCRPOS-40,SCRPOS-40)="c
" THEN RETURN
727 IF SCREEN$(SCRPOS-40,SCRPOS-40)="z
" THEN RETURN
730 IF SCREEN$(SCRPOS-40,SCRPOS-40)="D
" THEN SCREEN$(SCRPOS-40,SCRPOS-40)="z
":PLACED=PLACED+1
```

```
735 IF SCREEN$(SCRPOS-40,SCRPOS-40)="
" THEN SCREEN$(SCRPOS-40,SCRPOS-40)="c
"
750 IF SCREEN$(SCRPOS-20,SCRPOS-20)="z
" THEN SCREEN$(SCRPOS-20,SCRPOS-20)="v
":PLACED=PLACED-1:GOTO 760
755 SCREEN$(SCRPOS-20,SCRPOS-20)="a"
760 IF SCREEN$(SCRPOS,SCRPOS)="v" THEN
 SCREEN$(SCRPOS,SCRPOS)="d"
770 IF SCREEN$(SCRPOS,SCRPOS)="a" THEN
 SCREEN$(SCRPOS,SCRPOS)=" "
775 POSITION XPOS-1,YPOS+1:? #6;"o";:G
OSUB 1000
777 IF SCREEN$(SCRPOS,SCRPOS)="c" THEN
 POSITION XPOS-1,YPOS+1:? #6;"q";:GOTO
 781
780 POSITION XPOS-1,YPOS+1:? #6;"p";
781 POSITION XPOS-1,YPOS-1+1:? #6;"r"
:POSITION XPOS-1,YPOS-2+1:? #6;"s";
782 GOSUB 1000:POSITION XPOS-1,YPOS+1:
? #6;SCREEN$(SCRPOS,SCRPOS);:POSITION
XPOS-1,YPOS-1+1:? #6;"o";
783 POSITION XPOS-1,YPOS-2+1:? #6;"c";
:GOSUB 1000
784 POSITION XPOS-1,YPOS-1+1:? #6;"a";

785 PUSH=1:YPOS=YPOS-1:GOSUB 10700:RET
URN
800 IF SCREEN$(SCRPOS+20,SCRPOS+20)="B
" THEN RETURN
805 IF SCREEN$(SCRPOS+20,SCRPOS+20)="
" THEN GOTO 815
810 IF SCREEN$(SCRPOS+20,SCRPOS+20)="d
" THEN SCREEN$(SCRPOS+20,SCRPOS+20)="v
":GOTO 815
812 GOTO 820
815 IF SCREEN$(SCRPOS,SCRPOS)="a" THEN
 SCREEN$(SCRPOS,SCRPOS)=" "
817 IF SCREEN$(SCRPOS,SCRPOS)="v" THEN
 SCREEN$(SCRPOS,SCRPOS)="d"
818 POSITION XPOS-1,YPOS+1:? #6;"t";:F
OR I=1 TO 5:NEXT I:POSITION XPOS-1,YPO
S+1:? #6;SCREEN$(SCRPOS,SCRPOS);
819 POSITION XPOS-1,YPOS+1+1:? #6;"t";
:POSITION XPOS-1,YPOS+1+1:? #6;"a";:YP
OS=YPOS+1:GOSUB 1010:PUSH=0:RETURN
820 IF SCREEN$(SCRPOS+40,SCRPOS+40)="B
" THEN RETURN
825 IF SCREEN$(SCRPOS+40,SCRPOS+40)="c
" THEN RETURN
827 IF SCREEN$(SCRPOS+40,SCRPOS+40)="z
" THEN RETURN
830 IF SCREEN$(SCRPOS+40,SCRPOS+40)="d
" THEN SCREEN$(SCRPOS+40,SCRPOS+40)="z
":PLACED=PLACED+1
835 IF SCREEN$(SCRPOS+40,SCRPOS+40)="
" THEN SCREEN$(SCRPOS+40,SCRPOS+40)="c
"
850 IF SCREEN$(SCRPOS+20,SCRPOS+20)="z
" THEN SCREEN$(SCRPOS+20,SCRPOS+20)="v
":PLACED=PLACED-1:GOTO 860
855 SCREEN$(SCRPOS+20,SCRPOS+20)="a"
860 IF SCREEN$(SCRPOS,SCRPOS)="v" THEN
 SCREEN$(SCRPOS,SCRPOS)="d"
870 IF SCREEN$(SCRPOS,SCRPOS)="a" THEN
 SCREEN$(SCRPOS,SCRPOS)=" "
875 POSITION XPOS-1,YPOS+1:? #6;"t";:G
OSUB 1000
877 IF SCREEN$(SCRPOS,SCRPOS)="c" THEN
 POSITION XPOS-1,YPOS+1:? #6;"v";:GOTO
 881
880 POSITION XPOS-1,YPOS+1:? #6;"u";
881 POSITION XPOS-1,YPOS+1+1:? #6;"w";
:POSITION XPOS-1,YPOS+2+1:? #6;"x";
882 GOSUB 1000:POSITION XPOS-1,YPOS+1:
? #6;SCREEN$(SCRPOS,SCRPOS);:POSITION
XPOS-1,YPOS+1+1:? #6;"t";
883 POSITION XPOS-1,YPOS+2+1:? #6;"c";
:GOSUB 1000
884 POSITION XPOS-1,YPOS+1+1:? #6;"a";

885 PUSH=1:YPOS=YPOS+1:GOSUB 10700:RET
URN
900 POKE 106,PEEK(106)-8:GRAPHICS 0:CH
SET=PEEK(106)*256
902 IF PEEK(CHSET+(32+1)*8)=24 THEN RE
TURN
905 FOR I=0 TO 511:POKE CHSET+I,PEEK(5
7344+I):NEXT I
910 GRAPHICS 1:POKE 756,CHSET/256
915 FOR I=1 TO 26
917 FOR J=0 TO 7
919 READ A:POKE CHSET+((I+32)*8)+J,A
921 NEXT J
922 NEXT I
950 DATA 24,88,48,30,16,16,40,36
952 DATA 221,221,221,0,119,119,119,0
954 DATA 0,126,102,90,90,102,126,0
956 DATA 0,24,36,66,66,36,24,0
958 DATA 48,48,144,112,8,20,20,18
960 DATA 0,7,6,5,5,6,7,0
962 DATA 3,227,105,167,160,96,224,0
964 DATA 0,24,36,66,194,164,152,128
966 DATA 0,0,0,0,128,128,128,128
968 DATA 12,12,9,14,16,40,40,72
970 DATA 0,0,0,0,1,1,1,1
972 DATA 0,29,36,66,67,37,25,1
974 DATA 192,199,150,229,5,6,7,0
976 DATA 0,224,96,160,160,96,224,0
978 DATA 84,84,56,16,24,32,32,0
980 DATA 28,0,0,0,0,0,0,0
982 DATA 28,24,36,66,66,36,24,0
984 DATA 90,102,126,0,42,42,28,8
986 DATA 0,0,0,0,0,126,102,90
988 DATA 0,32,32,24,16,56,86,86
990 DATA 0,0,0,0,0,0,0,28
992 DATA 0,24,36,66,66,36,24,28
994 DATA 8,28,42,42,0,126,102,90
996 DATA 90,102,126,0,0,0,0,0
997 DATA 24,88,48,30,16,16,40,36
998 DATA 0,126,102,90,90,102,126,0
999 RETURN
1000 SOUND 0,PITCH+50,10,8:FOR W=1 TO
2:NEXT W:PITCH=30-PITCH:SOUND 0,0,0,0:
RETURN
1010 SOUND 0,50,10,8:SOUND 0,0,0,0:MOV
ES=MOVES+1:GOSUB 10750:RETURN
1080 FOR I=1 TO 80:A=INT(RND(0)*256):S
OUND 0,A,10,8:SETCOLOR 1,A,A/16:NEXT I
:SOUND 0,0,0,0:SETCOLOR 1,13,4
1085 TIME=HOURS*60*60+MINS*60+SECS
1090 IF MOVES<RM OR RM=0 OR PUSHES<RP
OR RP=0 OR TIME<RT OR RT=0 THEN GOTO 1
100
1095 GOTO 1150
1100 OPEN #1,8,0,FILNAM$:FOR I=1 TO 16
:? #1;BEGSCR$((I-1)*20+1,I*20):NEXT I
1105 ? #1;XP:? #1;YP:? #1;BO
1110 IF MOVES<RM OR RM=0 THEN ? #1;MOV
ES:? #1;NAM$:GOTO 1120
1115 ? #1;RM:? #1;RM$
1120 IF PUSHES<RP OR RP=0 THEN ? #1;PU
SHES:? #1;NAM$:GOTO 1130
1125 ? #1;RP:? #1;RP$
1130 IF TIME<RT OR RT=0 THEN ? #1;TIME
:? #1;NAM$:GOTO 1140
1135 ? #1;RT:? #1;RT$
1140 CLOSE #1
1150 RETURN
10500 GRAPHICS 18:POKE 16,64:POKE 5377
4,64:DL=PEEK(560)+256*PEEK(561):SETCOL
OR 2,0,0
10510 POKE DL+10,6:POKE DL+11,6:POKE D
L+12,6:POKE DL+13,2:SETCOLOR 0,8,4:POS
ITION 6,4
10520 ? #6;"SOKO-BAN":POSITION 5,8:? #
6;"ANSCHUETZ/WEISGERBER/ANSCHUETZ"
10530 POSITION 0,6:? #6;"press start t
o begin":IF PEEK(53279)<>6 THEN GOTO 1
0530
10540 RETURN
10600 TIME=INT((PEEK(18)*256*256+PEEK(
19)*256+PEEK(20))/60)
10605 HOURS=INT(TIME/60/60):TIME=TIME-
HOURS*60*60
10610 MINS=INT(TIME/60):TIME=TIME-MINS
*60
10615 SECS=INT(TIME)
10620 RETURN
10700 MOVES=MOVES+1:PUSHES=PUSHES+1:GO
SUB 10740:RETURN
10740 POSITION 15,19:IF PUSHES<10 THEN
 ? #6;"000";PUSHES;:GOTO 10750
10742 IF PUSHES<100 THEN ? #6;"00";PUS
HES;:GOTO 10750
10744 IF PUSHES<1000 THEN ? #6;"0";PUS
HES;:GOTO 10750
10746 ? #6;PUSHES;
10750 POSITION 1,19:IF MOVES<10 THEN ?
 #6;"000";MOVES;:GOTO 10760
10752 IF MOVES<100 THEN ? #6;"00";MOVE
S;:GOTO 10760
10754 IF MOVES<1000 THEN ? #6;"0";MOVE
S;:GOTO 10760
10756 ? #6;MOVES;
10760 RETURN
10800 IF LEVEL>MAXLEV THEN STAT=2:GOTO
 10860
10809 STAT=1:TEN$=CHR$(INT(LEVEL/10)+4
8):WON$=CHR$(LEVEL-INT(LEVEL/10)*10+48
)
10810 FILNAM$(1,8)="D:SCREEN":FILNAM$(
9,9)=TEN$:FILNAM$(10,10)=WON$:FILNAM$(
11,14)=".DAT"
10820 TRAP 10850:OPEN #1,4,0,FILNAM$
10825 FOR I=1 TO 16:INPUT #1;LINE$:SCR
EEN$((I-1)*20+1,I*20)=LINE$:NEXT I:INP
UT #1;XPOS:INPUT #1;YPOS
10827 INPUT #1;BOXES:INPUT #1;RM:INPUT
 #1;RM$:INPUT #1;RP:INPUT #1;RP$:INPUT
```

```
 #1;RT:INPUT #1;RT$
10828 XP=XPOS:YP=YPOS:BO=BOXES
10830 CLOSE #1:GOTO 10860
10850 CLOSE #1:? "DISK ERROR. PRESS RE
TURN TO CONTINUE";:TRAP 40000:STAT=2:T
RAP 10:INPUT X
10860 RETURN
10900 T1=PEEK(20):T2=PEEK(19):T3=PEEK(
18)
10905 GRAPHICS 2+16:POKE 16,64:POKE 53
774,64:POKE 752,1:POKE 756,224
10910 POSITION 0,0:? #6;"  soko-ban re
cords";:POSITION 4,1:? #6;"for level "
;:? #6;CHR$(INT(LEVEL/10)+48);
10911 ? #6;CHR$(LEVEL-INT(LEVEL/10)*10
+48);:POSITION 0,4:? #6;"moves   time
 pushes";
10912 POSITION 1,5:IF RM<10 THEN ? #6;
"000";RM;:GOTO 10920
10914 IF RM<100 THEN ? #6;"00";RM;:GOT
O 10920
10916 IF RM<1000 THEN ? #6;"0";RM;:GOT
O 10920
10918 ? #6;RM;
10920 TIME=RT:GOSUB 10605
10925 POSITION 6,5:? #6;"0";HOURS;":";
:IF MINS<10 THEN ? #6;"0";MINS;":";:GO
TO 10935
10930 ? #6;MINS;":";
10935 IF SECS<10 THEN ? #6;"0";SECS;:G
OTO 10945
10940 ? #6;SECS;
10945 POSITION 15,5:IF RP<10 THEN ? #6
;"000";RP;:GOTO 10960
10948 IF RP<100 THEN ? #6;"00";RP;:GOT
O 10960
10950 IF RP<1000 THEN ? #6;"0";RP;:GOT
O 10960
10955 ? #6;RP;
10960 POSITION 2,6:FOR I=1 TO 3:? #6;C
HR$(ASC(RM$(I,I))+128);:NEXT I
10962 POSITION 8,6:FOR I=1 TO 3:? #6;C
HR$(ASC(RT$(I,I))+128);:NEXT I
10964 POSITION 15,6:FOR I=1 TO 3:? #6;
CHR$(ASC(RP$(I,I))+128);:NEXT I
10970 POSITION 3,10:? #6;"press  trigg
er";
10980 IF STRIG(0)=1 THEN GOTO 10980
10990 GRAPHICS 1:POKE 16,64:POKE 53774
,64:POKE 752,1:POKE 18,T3:POKE 19,T2:P
OKE 20,T1:RETURN
```

# SOKO-BAN – SAVDAT

```
10 DIM SCREEN$(320),A$(20),B$(320),FIL
NAM$(15),RM$(3),RP$(3),RT$(3)
20 ? "ENTER FILENAME";:INPUT FILNAM$
45 OPEN #1,8,0,FILNAM$
47 GOSUB 1100
50 FOR I=1 TO 16
70 ? #1;SCREEN$((I-1)*20+1,I*20)
75 NEXT I
77 ? #1;XPOS:? #1;YPOS:? #1;BOXES
78 ? #1;0:? #1;"AAA":? #1;0:? #1;"AAA"
:? #1;0:? #1;"AAA"
80 CLOSE #1
81 XPOS=0:YPOS=0:BOXES=0
83 OPEN #1,4,0,FILNAM$
85 FOR I=1 TO 16
100 INPUT #1;A$
102 B$((I-1)*20+1,I*20)=A$
103 NEXT I
104 INPUT #1;XPOS:INPUT #1;YPOS:INPUT
#1;BOXES:INPUT #1;RM:INPUT #1;RM$:INPU
T #1;RP:INPUT #1;RP$:INPUT #1;RT
105 INPUT #1;RT$:CLOSE #1
110 FOR I=1 TO 16
120 ? B$((I-1)*20+1,I*20)
140 NEXT I
150 ? XPOS,YPOS,BOXES
160 END
1100 SCREEN$(1,20)="     BBBBBBB
  "
1105 SCREEN$(21,40)="    B   B  B
  "
1110 SCREEN$(41,60)="    B  c   B
  "
1115 SCREEN$(61,80)="  BBB Bc   BBBB
  "
1120 SCREEN$(81,100)="  B  c  BBc   B
  "
1125 SCREEN$(101,120)="  B  B a c B cB
  "
1130 SCREEN$(121,140)="  B  B      c B
BBB  "
1135 SCREEN$(141,160)="  BB BBBBcBB
  B  "
1140 SCREEN$(161,180)="  B cBDDDDDB B
  B  "
1145 SCREEN$(181,200)="  B  cDDzzD cB
BBB  "
1150 SCREEN$(201,220)=" BB  BDDDDDB
B    "
1155 SCREEN$(221,240)=" B   BBB BBBBBB
B    "
1160 SCREEN$(241,260)=" B cc  B  B
  "
1165 SCREEN$(261,280)=" B  B     B
  "
1170 SCREEN$(281,300)=" BBBBBB   B
  "
1175 SCREEN$(301,320)="      BBBBB
  "
1180 XPOS=8:YPOS=6:BOXES=15
1185 RETURN
```

# SPACE ASSAILANTS

```
100 ? "K":M=PEEK(106)-64:N=M*256:GOTO
440
110 SOUND 0,0,0,0:N5A=0:5E=5E+1:POKE 6
56,0:POKE 657,34:? 5E;"  ":IF LI=0 THE
N 290
120 FOR Y=13262+N TO 13271+N:POKE Y,0:
NEXT Y:POKE 54286,0
130 POKE 13273+N,50:LM=50:POKE 13163+N
,240:X=240:DR=13262+N:F=0
140 FOR Z=12859+N TO 13099+N STEP 4:PO
KE Z,0:POKE Z+1,0:NEXT Z
150 R=INT((5E-1)/6):NL=5E-R*6:IF 5E>6
THEN Q=0.5:IF 5E>12 THEN 5D=20:Q=0.25
160 RESTORE 790:FOR L1=13059+N TO 1309
5-NL*40+N STEP -40:READ TA:FOR L2=L1 T
O L1+20 STEP 4
170 POKE L2,TA:POKE L2+1,TA+1:N5A=N5A+
1:NEXT L2:NEXT L1:Q=ABS(Q):POKE 53278,
0:IF 5E>18 THEN Q=0.5
180 X=X-5D:POKE 13163+N,X:IF X=0 THEN
X=240
190 IF F=1 THEN R=R-7:GOSUB 310
200 IF X/80=INT(X/80) THEN POKE 54297,
CH:CH=(48+M)*(CH=60+M)+(60+M)*(CH=48+M
)
210 IF PEEK(54016)=251 THEN YP=YP-(YP>
53)*4:POKE PLX,YP
220 IF PEEK(54016)=247 THEN YP=YP+(YP<
185)*4:POKE PLX,YP
230 IF PEEK(53264)=0 AND F=0 THEN F=1:
R=204+(DR-(13262+N)):POKE PLX+1,YP+4:5
P=YP+4
240 IF F=1 THEN R=R-7:GOSUB 310
250 IF PEEK(53254)>1 OR PEEK(53255)>1
THEN Q=-Q:POKE DR,112:DR=DR+1:IF DR=13
272+N THEN 290
260 LM=LM+Q:POKE 13273+N,LM
270 IF PEEK(53252)>0 THEN 420
280 POKE 53278,0:GOTO 180
290 SETCOLOR 2,0,0:SETCOLOR 0,8,4:POKE
 54286,255:POKE 560,TRO1:POKE 561,TRO2
:FOR X=PLX TO PLX+3:POKE X,0:NEXT X
300 POKE 53278,0:POKE 84,0:? #6;"    "
:SOUND 2,0,0,0:SOUND 3,0,0,0:GOTO 670
310 IF PEEK(53253)>1 AND PEEK(53253)<8
 THEN JW=7:GOTO 350
320 POKE 88,R:POKE 84,0:? #6;"88\D
   ":IF R<50 THEN POKE 84,0:? #6;"
":F=0
330 IF PEEK(53253)>1 AND PEEK(53253)<8
 THEN JW=0:GOTO 350
340 RETURN
350 HP=12818+N+INT((((R-40)+JW-(DR-(13
262+N))*7)/14)+0.5)*40+INT(((5P-48)/16
+(LM-58)/4)*4)
360 IF HP>13090+N THEN RETURN
370 EA=0:FOR Z=HP TO HP+3:EA=EA+PEEK(Z
):POKE Z,0:NEXT Z:IF EA=0 THEN R=R+JW:
RETURN
380 POKE 84,0:? #6;"    "
390 SOUND 0,220,10,15
400 F=0:N5A=N5A-1:5C=5C+10:POKE 656,0:
POKE 657,18:? 5C;:IF N5A=0 THEN POP :G
OTO 110
410 SOUND 0,0,0,0:RETURN
420 LI=LI-1:POKE 656,0:POKE 657,5:? LI
;:POKE 84,0:? #6;"    ":5E=5E-1:FOR Z=
200 TO 255:SOUND 0,Z,8,15:NEXT Z
430 GOTO 110
440 GRAPHICS 2:TRO1=PEEK(560):TRO2=PEE
K(561):TRODL=TRO1+256*TRO2:SETCOLOR 2,
0,0:SETCOLOR 0,8,4
450 POKE TRODL+10,6:POKE TRODL+11,6:PO
KE TRODL+12,6:POKE TRODL+13,2:POKE 16,
64:POKE 53774,64
460 POSITION 2,4:? #6;"SPACE ASSAILANT
S":POSITION 5,8:? #6;"ANSCHUETZ/WEISGE
RBER/ANSCHUETZ"
470 POSITION 4,6:? #6;"initializing"
480 CH1=48*256+N:CH2=60*256+N:FOR I=0
TO 511:POKE CH1+I,PEEK(57344+I):POKE C
H2+I,PEEK(57344+I):NEXT I
490 FOR I=8 TO 55:READ A:POKE CH1+I,A:
NEXT I:FOR I=8 TO 55:READ A:POKE CH2+I
,A:NEXT I
500 FOR I=56 TO 79:READ A:POKE CH1+I,A
:POKE CH2+I,A:NEXT I:POKE 756,60+M:CH=
60+M
510 FOR MIAD=56*256+N TO 58*256+N:POKE
 MIAD,0:NEXT MIAD
520 FOR MIAD=53*256+N TO 54*256+N:POKE
 MIAD,0:NEXT MIAD
530 FOR M1=1 TO 10:M2=INT(RND(0)*220)+
14336+N:R=INT(RND(0)*7)+1:POKE M2,2^R:
POKE M2+240,2^R:NEXT M1
540 FOR I=13256+N TO 13291+N:READ A:IF
 A=50 OR A=62 OR A=51 THEN ZZ=ZZ+1:IF
ZZ>1 THEN A=A+M
550 IF A=50 OR A=62 OR A=51 THEN ZZ=ZZ
+1:IF ZZ>1 THEN A=A+M
560 POKE I,A:NEXT I:FOR I=13156+N TO 1
3165+N:READ A:A=A+M*(A=56):POKE I,A:NE
XT I
570 FOR I=13166+N TO 13210+N STEP 2:PO
KE I,9:POKE I+1,48:NEXT I
580 FOR I=13211+N TO 13213+N:READ A:A=
A+M*(A=51):POKE I,A:NEXT I:DIM A$(20):
A$="SPACE ASSAILANTS"
590 POKE 13260+N,PEEK(660):POKE 13261+
N,PEEK(661):POKE 13160+N,PEEK(660):POK
E 13161+N,PEEK(661)
600 PM=48*256+N:PLX=53248:FOR X=216 TO
 223:READ A:POKE PM+1024+X,A:NEXT X
610 FOR X=PM+1536 TO PM+2047:POKE X,25
5:NEXT X
620 FOR X=1 TO 16:A=PEEK(ADR(A$(X)))-3
2:FOR Y=0 TO 7:POKE X*8+Y+PM+1576,255-
PEEK(A*8+Y+57344)
630 POKE X*8+Y+PM+1832,255-PEEK(A*8+Y+
57344):NEXT Y:NEXT X:POKE 704,50:POKE
705,40:POKE 706,130:POKE 707,130
640 POKE 559,62:POKE 623,1:POKE 53277,
3:POKE 54279,PM/256:POKE 53256,1
650 POKE 15880+N,199:POKE 15881+N,200:
POKE 15882+N,200:POKE 15883+N,201
660 POSITION 0,6:? #6;"press start to
begin":POKE 89,53+M:POKE 82,0:POKE 755
,0
670 FOR RA=12859+N TO 13099+N STEP 4:P
OKE RA,0:POKE RA+1,0:NEXT RA
680 IF PEEK(53279)<>6 THEN 680
690 POKE 560,100:POKE 561,51+M:SETCOLO
R 0,12,8:SETCOLOR 1,3,6:SETCOLOR 2,8,2
:SETCOLOR 3,12,4
700 POKE PLX+2,48:POKE PLX+3,200:POKE
PLX,121:YP=121:Q=0.25:5E=0:5C=0:N5A=0:
5D=10:LI=3
710 POKE 656,0:POKE 657,5:? LI;:POKE 6
57,18:? 5C;"      ":SOUND 2,190,10,3:5
OUND 3,192,10,3:GOTO 110
720 DATA 3,15,59,63,15,4,16,65,192,240
,236,252,240,16,64,0,0,67,79,30,14,14,
15,20
730 DATA 0,193,241,180,176,176,240,20,
0,4,19,63,62,30,79,16,0,16,196,252,188
,180,241,4
740 DATA 3,15,62,63,15,4,1,0,192,240,1
88,252,240,16,4,65,0,3,15,15,30,79,79,
5
750 DATA 0,192,240,240,180,241,241,80,
16,4,3,63,62,30,31,20,4,16,192,252,188
,180,244,20
760 DATA 0,0,0,0,11,63,127,227,0,0,60,
255,153,153,255,24,0,0,0,0,208,252,254
,199,112,112,112,66,0,0,0,0,0,0,0,0,0
770 DATA 0,0,0,0,68,50,50,96,4,96,4,96
,4,96,4,96,4,112,70,0,62,65,100,51,112
,112,112,66,0,0,73,0,56,48,65,200,51
780 DATA 8,8,28,28,62,62,42,42
790 DATA 1,3,5,1,3,5
```

# YAHTZEE

```
100 GOSUB 2510:DIM NAME1$(16),NAME2$(1
6),NAME3$(16),NAME4$(16),NAME5$(16),BL
KNM$(16):BLKNM$(1)=" ":BLKNM$(16)=" "
110 BLKNM$(2)=BLKNM$
120 DIM SCORES(19,5),ROLL(5),OCC(6),AN
S$(1),CHANGE(5),TURN$(5)
130 GOSUB 1570:GOTO 150
140 ? CHR$(125):POKE 752,1:FOR I=1 TO
18:FOR J=1 TO 5:SCORES(I,J)=0:TURN$(J)
=0:NEXT J:NEXT I:PL=1:RETURN
150 GOSUB 140:GOSUB 2210:GOSUB 1360
160 GOTO 300
170 POSITION 0,0:? WON$:RETURN
180 POSITION 0,0:? TWO$:RETURN
190 POSITION 0,0:? THREE$:RETURN
200 POSITION 0,0:? FOUR$:RETURN
210 POSITION 0,0:? FIVE$:RETURN
220 POSITION 0,0:? SIX$:RETURN
230 POSITION 2,2:? CHR$(17);:POSITION
2,3:? CHR$(124);:POSITION 2,4:? CHR$(2
6);:POSITION 3,3:? "YAHTZEE";
240 FOR I=3 TO 9:POSITION I,2:? CHR$(1
8);:POSITION I,4:? CHR$(18);:NEXT I:PO
SITION 10,2:? CHR$(5);
250 POSITION 10,3:? CHR$(124);:POSITIO
N 10,4:? CHR$(3);
260 RESTORE 2620:FOR I=1 TO 9:FOR J=1
TO 7:READ A:POSITION J+2,I+8:? CHR$(A)
;:NEXT J:NEXT I:RETURN
270 POKE 89,INT(PL1/256)+2:POKE 88,Y
280 POSITION 0,0:? BLK16$;:POKE 88,NY:
POSITION 0,0:? LEFT$
290 POKE 89,U:POKE 88,L:Y=NY:RETURN
300 ROLLS=1:FOR I=1 TO 5:ROLL(I)=INT(R
ND(0)*6)+1:NEXT I
310 POKE 89,INT(PL1/256):FOR I=1 TO 15
:YY=INT(RND(0)*5)+1:POKE 88,YY*40-8
320 EA=INT(RND(0)*6)+1:ON EA GOSUB 170
,180,190,200,210,220:NEXT I
330 POKE 89,INT(PL1/256):FOR I=1 TO 5
340 POKE 88,I*40-8:ON ROLL(I) GOSUB 17
0,180,190,200,210,220:NEXT I
350 POKE 89,U:POKE 88,L:POSITION 17,23
:? "     FINISHED?(Y/N)";:POKE 764,255
360 GOSUB 580
370 IF PEEK(764)=43 THEN GOTO 610
380 POSITION 17,23:? "
  ";:POKE 89,INT(PL1/256)+2:POKE 88,Y:
POSITION 0,0:? BLK16$
390 POKE 764,255:Y=40:POKE 89,INT(PL1/
256)+2:POKE 88,Y:POSITION 0,0:? LEFT$
400 ST=STICK(0)
410 IF ST=14 THEN NY=Y-40:NY=NY+200*(N
Y=0):GOSUB 270
420 IF ST=13 THEN NY=Y+40:NY=NY-200*(N
Y=240):GOSUB 270
430 IF STRIG(0) THEN GOTO 480
440 POKE 89,U:POKE 88,L
450 IF CHANGE(Y/40)=0 THEN POSITION 11
,(Y/40-1)*5+1:? CHR$(17);">";:CHANGE(Y
/40)=1:GOTO 470
460 POSITION 11,(Y/40-1)*5+1:? "  ";:C
HANGE(Y/40)=0
470 FOR DELAY=1 TO 25:NEXT DELAY
480 IF PEEK(764)=33 THEN 500
490 GOTO 400
500 FOR I=1 TO 5:POSITION 11,(I-1)*5+1
:? "  ";:NEXT I:FOR I=1 TO 5
510 IF  NOT (CHANGE(I)) THEN GOTO 550
520 CHANGE(I)=0:POKE 89,INT(PL1/256):P
OKE 88,(I-1)*40+32
530 FOR J=1 TO 5:BP=INT(RND(0)*6)+1:ON
 BP GOSUB 170,180,190,200,210,220:NEXT
 J
540 ROLL(I)=INT(RND(0)*6)+1:ON ROLL(I)
 GOSUB 170,180,190,200,210,220
550 NEXT I
560 ROLLS=ROLLS+1:IF ROLLS=3 THEN 610
570 GOTO 330
580 YUNHUI=0
590 IF PEEK(764)<>255 THEN POSITION 36
,23:? " ";:SOUND 0,50,12,8:FOR I=1 TO
20:NEXT I:SOUND 0,0,0,0:RETURN
600 YUNHUI=1-YUNHUI:POSITION 36,23:? C
HR$(32+63*YUNHUI);:FOR I=1 TO 30:NEXT
I:GOTO 590
610 POKE 89,INT(PL1/256)+2:POKE 88,Y:P
OSITION 0,0:? BLK16$
620 POKE 89,U:POKE 88,L:POSITION 17,23
:? "     CHOOSE OPTION:";
630 POKE 53258,0:POKE 53250,190
640 POKE 764,255:Y=50:POKE 89,INT(PL1/
256)+2:POKE 88,Y:POSITION 0,0:? RIGHT$

650 ST=STICK(0)
660 IF ST=14 THEN NY=Y-8:NY=NY+136*(NY
=42)-24*(NY=114)-8*(NY=170):GOSUB 700
670 IF ST=13 THEN NY=Y+8:NY=NY-136*(NY
=186)+24*(NY=98)+8*(NY=170):GOSUB 700
680 IF STRIG(0)=0 THEN GOTO 730
690 GOTO 650
700 POKE 89,INT(PL1/256)+2:POKE 88,Y
710 POSITION 0,0:? BLK8$;:POKE 88,NY:P
OSITION 0,0:? RIGHT$
720 POKE 89,U:POKE 88,L:Y=NY:RETURN
730 POKE 89,U:POKE 88,L:POKE 764,255:C
HOICE=(Y-50)/8+1
740 ON CHOICE GOSUB 1690,1710,1730,175
0,1770,1790,0,0,0,1810,1850,1890,1970,
2040,2110,0,2180
750 IF CHOICE=15 AND SCORES(16,PL)>0 A
ND OK THEN GOTO 870
760 IF SCORES(CHOICE,PL)<>0 THEN GOSUB
 1350:POSITION 17,23:? "       TRY AG
AIN!";:FOR I=1 TO 100:NEXT I:GOTO 620
770 IF OK THEN GOTO 840
780 GOSUB 1350:POSITION 17,23:? "NIL T
HAT CATAGORY?";
790 IF PEEK(764)=255 THEN GOTO 780
800 SOUND 0,50,12,8:FOR I=1 TO 20:NEXT
 I:SOUND 0,0,0,0
810 IF PEEK(764)=43 THEN SCORES(CHOICE
,PL)=-1:POSITION 36,CHOICE+1:? "NIL";:
GOTO 860
820 IF PEEK(764)=35 THEN GOTO 620
830 POKE 764,255:GOTO 790
840 SCORES(CHOICE,PL)=TOT
850 POSITION 36,CHOICE+1:? SCORES(CHOI
CE,PL);
860 TURN$(PL)=TURN$(PL)+1
870 DONE=1:FOR I=1 TO NUMPL:IF TURN$(I
)<>13 THEN DONE=0
880 NEXT I:IF DONE THEN GOTO 1090
890 GOSUB 900:POKE 53250,120:GOTO 300
900 POKE 89,INT(PL1/256)+2:POKE 88,Y:P
OSITION 0,0:? BLK8$:POKE 89,U:POKE 88,
L
910 POSITION 17,23:? "PRESS FIRE BUTTO
N:";
920 IF STRIG(0) THEN GOTO 920
930 POSITION 17,23:? "
  ";
940 PL=PL+1:IF PL>NUMPL THEN PL=1
950 IF TURN$(PL)=13 THEN GOTO 940
960 POSITION 19,0:? BLKNM$;:ON PL GOSU
B 1040,1050,1060,1070,1080
970 FOR I=1 TO 6:GOSUB 980:NEXT I:FOR
I=10 TO 17:GOSUB 980:NEXT I:GOTO 1010
980 POSITION 36,I+1:? "   ";:POSITION
36,I+1:IF SCORES(I,PL)>0 THEN ? SCORES
(I,PL);:RETURN
990 IF SCORES(I,PL)=-1 THEN ? "NIL";:R
ETURN
1000 RETURN
1010 POSITION 27,17:? CHR$(20);" ";CHR
$(20);" ";CHR$(20);" ";CHR$(20);":";
1020 IF SCORES(16,PL)=0 THEN RETURN
1030 FOR I=1 TO SCORES(16,PL)/100:POSI
TION 27+(I-1)*2,17:? CHR$(148);:NEXT I
:RETURN
1040 POSITION 35-LEN(NAME1$),0:? NAME1
$;:RETURN
1050 POSITION 35-LEN(NAME2$),0:? NAME2
$;:RETURN
1060 POSITION 35-LEN(NAME3$),0:? NAME3
$;:RETURN
1070 POSITION 35-LEN(NAME4$),0:? NAME4
$;:RETURN
1080 POSITION 35-LEN(NAME5$),0:? NAME5
$;:RETURN
1090 POKE 89,INT(PL1/256)+2:POKE 88,Y:
POSITION 0,0:? BLK8$:POKE 89,U:POKE 88
,L
1100 FOR PL=1 TO NUMPL
1110 POSITION 17,23:? "PRESS FIRE BUTT
ON:";
1120 IF STRIG(0) THEN GOTO 1120
1130 POSITION 19,0:? BLKNM$;:ON PL GOS
UB 1040,1050,1060,1070,1080
1140 TOT=0:FOR I=1 TO 6:TOT=TOT+SCORES
(I,PL)*(SCORES(I,PL)<>-1):POSITION 36,
I+1:? "   ";:POSITION 36,I+1
1150 IF SCORES(I,PL)=-1 THEN ? "NIL";:
GOTO 1170
1160 ? SCORES(I,PL);
1170 NEXT I:IF TOT>=63 THEN SCORES(7,P
L)=35:POSITION 36,8:? "35";:GOTO 1190
1180 SCORES(7,PL)=0:POSITION 36,8:? "0
 ";
1190 TOT=TOT+SCORES(7,PL):SCORES(8,PL)
=TOT:POSITION 36,9:? "   ";:POSITION 3
6,9:? TOT;
1200 TOT=0:FOR I=10 TO 17:TOT=TOT+SCOR
ES(I,PL)*(SCORES(I,PL)<>-1):POSITION 3
6,I+1:? "   ";:POSITION 36,I+1
1210 IF SCORES(I,PL)=-1 THEN ? "NIL";:
GOTO 1230
1220 ? SCORES(I,PL);
1230 NEXT I:SCORES(18,PL)=TOT:POSITION
 36,19:? "   ";:POSITION 36,19:? TOT;
```

```
1240 POSITION 27,17:? CHR$(20);" ";CHR
$(20);" ";CHR$(20);" ";CHR$(20);":";
1250 IF SCORES(16,PL)=0 THEN POSITION
36,17:? "    ";:GOTO 1270
1260 FOR I=1 TO SCORES(16,PL)/100:POSI
TION 27+(I-1)*2,17:? CHR$(148);:NEXT I

1270 SCORES(19,PL)=SCORES(8,PL)+SCORES
(18,PL):POSITION 36,21:? "   ";:POSITI
ON 36,21:? SCORES(19,PL);
1280 NEXT PL
1290 POKE 764,255:POSITION 17,23:? "
     PLAY AGAIN?";
1300 IF PEEK(764)=255 THEN GOTO 1300
1310 SOUND 0,50,12,8:FOR I=1 TO 20:NEX
T I:SOUND 0,0,0,0:FOR I=0 TO 3
1320 POKE 53248+I,0:NEXT I:IF PEEK(764
)<>43 THEN ? CHR$(125):POKE 764,255:PO
KE 752,0:POKE 54286,64:END
1330 GOSUB 1570:GOSUB 140:GOSUB 1360
1340 POKE 53248,105:POKE 53249,105:POK
E 53250,120:POKE 53258,2:GOTO 300
1350 POKE 89,INT(PL1/256)+2:POKE 88,Y:
POSITION 0,0:? BLK8$:POKE 89,U:POKE 88
,L:RETURN
1360 ? CHR$(125):GOSUB 230:POKE 1599,0
:X=USR(1625):POKE 54286,192
1370 POSITION 35-LEN(NAME1$),0:? NAME1
$;
1380 POSITION 30,2:? "ONES:     ";
1390 POSITION 30,3:? "TWOS:     ";
1400 POSITION 28,4:? "THREES:    ";
1410 POSITION 29,5:? "FOURS:     ";
1420 POSITION 29,6:? "FIVES:     ";
1430 POSITION 29,7:? "SIXES:     ";
1440 POSITION 29,8:? "BONUS:     ";
1450 POSITION 23,9:? "UPPER TOTAL:
";
1460 POSITION 23,11:? "3 OF A KIND:
 ";
1470 POSITION 23,12:? "4 OF A KIND:
 ";
1480 POSITION 24,13:? "FULL HOUSE:
";
1490 POSITION 20,14:? "SMALL STRAIGHT:
   ";
1500 POSITION 20,15:? "LARGE STRAIGHT:
  ";
1510 POSITION 27,16:? "YAHTZEE:    ";
1520 POSITION 27,17:? CHR$(20);" ";CHR
$(20);" ";CHR$(20);" ";CHR$(20);":
";
1530 POSITION 28,18:? "CHANCE:     ";
1540 POSITION 23,19:? "LOWER TOTAL:
 ";
1550 POSITION 29,21:? "TOTAL:     ";
1560 RETURN
1570 POKE 54286,64:POKE 764,255:? CHR$
(125):POKE 709,54:POKE 710,50:POKE 712
,132:POKE 1599,0:TRAP 40000
1580 ? CHR$(125):POKE 752,0:TRAP 40000

1590 TRAP 1580:POSITION 10,10:? "HOW M
ANY PLAYERS";:INPUT NUMPL
1600 IF NUMPL<1 OR NUMPL>5 THEN GOTO 1
580
1610 TRAP 40000:TRAP 1610:FOR I=1 TO N
UMPL:? CHR$(125):POSITION 6,10:? "NAME
 OF PLAYER ";I;
1620 ON I GOSUB 1640,1650,1660,1670,16
80:NEXT I
1630 TRAP 40000:POKE 752,1:RETURN
1640 INPUT NAME1$:RETURN
1650 INPUT NAME2$:RETURN
1660 INPUT NAME3$:RETURN
1670 INPUT NAME4$:RETURN
1680 INPUT NAME5$:RETURN
1690 TOT=0:OK=0:FOR I=1 TO 5:TOT=TOT+(
ROLL(I)=1):NEXT I:IF TOT THEN OK=1
1700 RETURN
1710 TOT=0:OK=0:FOR I=1 TO 5:TOT=TOT+2
*(ROLL(I)=2):NEXT I:IF TOT THEN OK=1
1720 RETURN
1730 TOT=0:OK=0:FOR I=1 TO 5:TOT=TOT+3
*(ROLL(I)=3):NEXT I:IF TOT THEN OK=1
1740 RETURN
1750 TOT=0:OK=0:FOR I=1 TO 5:TOT=TOT+4
*(ROLL(I)=4):NEXT I:IF TOT THEN OK=1
1760 RETURN
1770 TOT=0:OK=0:FOR I=1 TO 5:TOT=TOT+5
*(ROLL(I)=5):NEXT I:IF TOT THEN OK=1
1780 RETURN
1790 TOT=0:OK=0:FOR I=1 TO 5:TOT=TOT+6
*(ROLL(I)=6):NEXT I:IF TOT THEN OK=1
1800 RETURN
1810 TOT=0:OK=0:FOR I=1 TO 6:OCC(I)=0:
NEXT I:FOR I=1 TO 5:OCC(ROLL(I))=OCC(R
OLL(I))+1:NEXT I
1820 FOR I=1 TO 6:TOT=TOT+I*OCC(I):IF
OCC(I)>=3 THEN OK=1
1830 NEXT I
1840 RETURN
1850 TOT=0:OK=0:FOR I=1 TO 6:OCC(I)=0:
NEXT I:FOR I=1 TO 5:OCC(ROLL(I))=OCC(R
OLL(I))+1:NEXT I
1860 FOR I=1 TO 6:TOT=TOT+I*OCC(I):IF
OCC(I)>=4 THEN OK=1
1870 NEXT I
1880 RETURN
1890 TOT=0:OK=0:OK2=0:OK3=0:FOR I=1 TO
 6:OCC(I)=0:NEXT I:FOR I=1 TO 5:OCC(RO
LL(I))=OCC(ROLL(I))+1:NEXT I
1900 FOR I=1 TO 6:IF OCC(I)=2 THEN IF
OK3=1 THEN OK=1:GOTO 1940
1910 IF OCC(I)=2 THEN OK2=1:GOTO 1940
1920 IF OCC(I)=3 THEN IF OK2=1 THEN OK
=1:GOTO 1940
1930 IF OCC(I)=3 THEN OK3=1
1940 NEXT I
1950 IF OK THEN TOT=25
1960 RETURN
1970 TOT=0:OK=0:FOR I=1 TO 6:OCC(I)=0:
NEXT I:FOR I=1 TO 5:OCC(ROLL(I))=OCC(R
OLL(I))+1:NEXT I
1980 COUNT=0:FOR I=1 TO 6:IF OCC(I) TH
EN COUNT=COUNT+1:GOTO 2010
1990 IF COUNT>=4 THEN OK=1
2000 COUNT=0
2010 NEXT I:IF COUNT>=4 THEN OK=1
2020 IF OK THEN TOT=30
2030 RETURN
2040 TOT=0:OK=0:FOR I=1 TO 6:OCC(I)=0:
NEXT I:FOR I=1 TO 5:OCC(ROLL(I))=OCC(R
OLL(I))+1:NEXT I
2050 COUNT=0:FOR I=1 TO 6:IF OCC(I) TH
EN COUNT=COUNT+1:GOTO 2080
2060 IF COUNT>=5 THEN OK=1
2070 COUNT=0
2080 NEXT I:IF COUNT>=5 THEN OK=1
2090 IF OK THEN TOT=40
2100 RETURN
2110 TOT=0:OK=1:NUM=ROLL(1):FOR I=2 TO
 5:IF ROLL(I)<>NUM THEN OK=0
2120 NEXT I
2130 IF OK THEN TOT=50:GOSUB 2190
2140 IF SCORES(15,PL)<>50 THEN GOTO 21
70
2150 SCORES(16,PL)=SCORES(16,PL)+100:P
OSITION 27+(INT(SCORES(16,PL)/100)-1)*
2,17:? CHR$(148);
2160 POSITION 36,17:? SCORES(16,PL);
2170 RETURN
2180 TOT=0:OK=1:FOR I=1 TO 5:TOT=TOT+R
OLL(I):NEXT I:RETURN
2190 POSITION 5,14:? CHR$(7);CHR$(14);
CHR$(6);:FOR I=1 TO 20:NEXT I:POSITION
 7,12:? CHR$(20);
2200 FOR I=1 TO 50:NEXT I:POSITION 7,1
2:? CHR$(111);:POSITION 5,14:? CHR$(32
);CHR$(18);CHR$(32);:RETURN
2210 PM=PEEK(106)-16:PL1=PM*256+1024:P
OKE 559,62:POKE 623,1:POKE 53256,2:POK
E 53258,2:POKE 53277,3:POKE 54279,PM
2220 POKE 704,0:POKE 705,44:POKE 706,3
2:POKE 53248,105:POKE 53249,105:POKE 5
3250,120
2230 DIM WON$(32),TWO$(32),THREE$(32),
FOUR$(32),FIVE$(32),SIX$(32):U=PEEK(89
):L=PEEK(88):POKE 89,INT(PL1/256)
2240 DIM BLANK$(32):BLANK$(1)=CHR$(32)
:BLANK$(32)=CHR$(32):BLANK$(2)=BLANK$
2250 FOR I=1 TO 5:FOR J=PL1+256+I*40-8
 TO PL1+256+I*40-8+31:POKE J,255:NEXT
J:NEXT I
2260 DIM BLK8$(8),BLK16$(16):BLK8$(1)=
CHR$(32):BLK8$(8)=CHR$(32):BLK8$(2)=BL
K8$
2270 BLK16$(1,8)=BLK8$:BLK16$(9,16)=BL
K8$
2280 DIM LEFT$(16),RIGHT$(8):FOR I=1 T
O 8:READ A:LEFT$(I*2-1,I*2-1)=CHR$(A):
LEFT$(I*2,I*2)=CHR$(A):NEXT I
2290 FOR I=1 TO 8:READ A:RIGHT$(I,I)=C
HR$(A):NEXT I
2300 FOR I=1 TO 8:READ A:FOR J=1 TO 4
2310 WON$((I-1)*4+J,(I-1)*4+J)=CHR$(A)

2320 NEXT J:NEXT I
2330 FOR I=1 TO 8:READ A:FOR J=1 TO 4
2340 TWO$((I-1)*4+J,(I-1)*4+J)=CHR$(A)

2350 NEXT J:NEXT I
2360 FOR I=1 TO 8:READ A:FOR J=1 TO 4
2370 THREE$((I-1)*4+J,(I-1)*4+J)=CHR$(
A)
2380 NEXT J:NEXT I
2390 FOR I=1 TO 8:READ A:FOR J=1 TO 4
2400 FOUR$((I-1)*4+J,(I-1)*4+J)=CHR$(A
)
2410 NEXT J:NEXT I
```

```
2420 FOR I=1 TO 8:READ A:FOR J=1 TO 4
2430 FIVE$((I-1)*4+J,(I-1)*4+J)=CHR$(A
)
2440 NEXT J:NEXT I
2450 FOR I=1 TO 8:READ A:FOR J=1 TO 4
2460 SIX$((I-1)*4+J,(I-1)*4+J)=CHR$(A)

2470 NEXT J:NEXT I:POKE 54286,64
2480 DL=PEEK(560)+256*PEEK(561):POKE D
L+8,130:POKE DL+13,130:POKE DL+18,130:
POKE DL+23,130:POKE 512,0:POKE 513,6
2490 FOR I=1536 TO 1591:READ A:POKE I,
A:NEXT I:FOR I=1600 TO 1611:READ A:POK
E I,A:NEXT I
2500 FOR I=1625 TO 1635:READ A:POKE I,
A:NEXT I:POKE 89,U:POKE 88,L:RETURN
2510 GRAPHICS 1:SETCOLOR 2,0,0:SETCOLO
R 0,8,4:POKE 16,64:POKE 53774,64:DL=PE
EK(560)+256*PEEK(561):POKE DL+14,23
2520 POKE DL+18,2:POKE 54276,4:POSITIO
N 8,9:? #6;"YAHTZEE":POSITION 4,11:? #
6;"press start to begin"
2530 POSITION 9,13:? #6;"ANSCHUETZ/WEI
SGERBER/ANSCHUETZ"
2540 IF PEEK(53279)<>6 THEN GOTO 2540
2550 GRAPHICS 0:POKE 16,64:POKE 53774,
64:RETURN
2560 DATA 64,96,248,108,70,34,34,34,64
,96,248,108,70,34,34,34
2570 DATA 32,32,32,56,56,32,32,32,35,3
5,32,32,32,32,128,128,35,35,32,56,56,3
2,128,128
2580 DATA 131,131,32,32,32,32,131,131,
131,131,32,56,56,32,131,131,131,131,32
,131,131,32,131,131
2590 DATA 72,138,72,152,72,174,63,6,18
9,64,6,168,232,189,64,6,141,10,212,141
,19,208,140,24,208,232
2600 DATA 189,64,6,141,23,208,232,224,
12,208,2,162,0,142,63,6,104,168,104,17
0,104,64,169,0,141,63,6,76,98,228
2610 DATA 50,172,54,138,12,132,138,60,
132,50,140,54,104,169,7,162,6,160,48,3
2,92,228,96
2620 DATA 21,21,21,21,21,21,21,32,32,1
24,32,124,32,32,32,6,13,13,13,7,32,32,
22,111,32,111,2,32,64,22,32,62
2630 DATA 32,2,64,32,22,32,18,32,2,32,
32,7,14,14,14,6,32,32,6,32,32,32,7,32,
6,32,32,123,32,32,7
```